Attorney Docket No. 1095.1286

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:

Ryoko FUJIKAWA, et al.

Application No.: Unassigned | Group Art Unit: Unassigned

Filed: September 25, 2003 | Examiner: Unassigned

For: PROGRAM AND PROCESS FOR GENERATING DATA USED IN SOFTWARE FUNCTION TEST

SUBMISSION OF CERTIFIED COPY OF PRIOR FOREIGN APPLICATION IN ACCORDANCE WITH THE REQUIREMENTS OF 37 C.F.R. § 1.55

Commissioner for Patents
PO Box 1450
Alexandria, VA 22313-1450

Sir:

In accordance with the provisions of 37 C.F.R. § 1.55, the applicants submit herewith a certified copy of the following foreign application:

Japanese Patent Application No. 2002-278806

Filed: September 25, 2002

It is respectfully requested that the applicants be given the benefit of the foreign filing date as evidenced by the certified papers attached hereto, in accordance with the requirements of 35 U.S.C. § 119.

Respectfully submitted,

STAAS & HALSEY LLP

Date: September 25, 2003 By: _____

William F. Herbert
Registration No. 31,024

1201 New York Ave, N.W., Suite 700
Washington, D.C. 20005
Telephone: (202) 434-1500
Facsimile: (202) 434-1501

日　本　国　特　許　庁
JAPAN　PATENT　OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出願年月日
Date of Application:　２００２年　９月２５日

出願番号
Application Number:　特願２００２－２７８８０６

[ST.10/C]:　[ＪＰ２００２－２７８８０６]

出願人
Applicant(s):　富士通株式会社

２００３年　３月　４日

特許庁長官
Commissioner,
Japan Patent Office　太田信一郎

出証番号　出証特２００３－３０１３４９７

【書類名】　特許願

【整理番号】　0252045

【提出日】　平成14年 9月25日

【あて先】　特許庁長官殿

【国際特許分類】　G06F 11/28

【発明の名称】　ソフトウェア機能テストデータ生成プログラムおよびソフトウェア機能テストデータ生成方法

【請求項の数】　5

【発明者】

【住所又は居所】　神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

【氏名】　藤川　亮子

【発明者】

【住所又は居所】　神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

【氏名】　伊藤　吾朗

【発明者】

【住所又は居所】　大分県大分市東春日町１７番５８号　株式会社富士通大分ソフトウェアラボラトリ内

【氏名】　片山　徹也

【発明者】

【住所又は居所】　静岡県静岡市南町１８番１号　株式会社富士通インフォソフトテクノロジ内

【氏名】　冨永　泰伸

【発明者】

【住所又は居所】　静岡県静岡市南町１８番１号　株式会社富士通インフォソフトテクノロジ内

【氏名】　川島　悟

【発明者】

【住所又は居所】　静岡県静岡市南町１８番１号　株式会社富士通インフォソフトテクノロジ内

【氏名】　齋藤　斎

【発明者】

【住所又は居所】　静岡県静岡市南町１８番１号　株式会社富士通インフォソフトテクノロジ内

【氏名】　長谷川　真則

【特許出願人】

【識別番号】　000005223

【氏名又は名称】　富士通株式会社

【代理人】

【識別番号】　100092152

【弁理士】

【氏名又は名称】　服部　毅巖

【電話番号】　0426-45-6644

【手数料の表示】

【予納台帳番号】　009874

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【包括委任状番号】　9705176

【プルーフの要否】　要

【書類名】　　明細書

【発明の名称】　ソフトウェア機能テストデータ生成プログラムおよびソフトウェア機能テストデータ生成方法

【特許請求の範囲】

【請求項１】　開発対象プログラムで定義された機能に対して入力するパラメタと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するためのテスト支援プログラムにおいて、

コンピュータに、

前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述を取得し、

更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述から、前記更新後動作記述と共通性の高い前記更新前動作記述を選択し、

更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターンから、選択された前記更新前動作記述の動作テストのための前記更新前テストパターンを抽出し、

抽出した前記更新前テストパターンの少なくとも一部を引き継いで、前記更新後動作記述の動作テストのための更新後テストパターンを生成する、

処理を実行させることを特徴とするテスト支援プログラム。

【請求項２】　前記更新後動作記述と選択された前記更新前動作記述とで共通した情報の属性に応じて、抽出した前記更新前テストパターンから引き継ぐべき情報を決定することを特徴とする請求項１記載のテスト支援プログラム。

【請求項３】　前記更新後テストパターンのうち、前記更新前テストパターンから引き継がない情報を自動生成することを特徴とする請求項１記載のテスト支援プログラム。

【請求項４】　共通性の高い前記更新前動作記述がない場合、操作入力により指定された前記更新前動作記述を選択することを特徴とする請求項１記載のテスト支援プログラム。

【請求項５】　開発対象プログラムで定義された機能に対して入力するパラ

メタと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するためのテスト支援方法において、

前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述を取得し、

更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述から、前記更新後動作記述と共通性の高い前記更新前動作記述を選択し、

更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターンから、選択された前記更新前動作記述の動作テストのための前記更新前テストパターンを抽出し、

抽出した前記更新前テストパターンの少なくとも一部を引き継いで、前記更新後動作記述の動作テストのための更新後テストパターンを生成する、

ことを特徴とするテスト支援方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は開発中のソフトウェアの機能を検証するための動作テストを支援するためのテスト支援プログラムおよびテスト支援方法に関し、特に機能に対して入力するパラメタと機能からの戻り値とを含むテストパターンを用いてテストを行う際のテストパターンの生成を支援するテスト支援プログラムおよびテスト支援方法に関する。

【０００２】

【従来の技術】

大規模なソフトウェアの開発では、最初から不具合（バグ）の全くないプログラムを作成するのはほとんど不可能である。そのため、ソフトウェアを開発する際には、作成したプログラムが設計通りに機能することをテストする必要がある。完成前のテストによって不具合を発見し、適宜プログラムを修正すことにより、設計通りに正しく動作するソフトウェアを開発することが出来る。

【０００３】

たとえば、オブジェクト指向で作成されたプログラムでは、機能毎にメソッドが作成される。メソッドは、入力データに応じて所定の処理を実行し、戻り値を出力する。そこで、オブジェクト指向のプログラムのテストでは、メソッドに対して入力するデータ（パラメタ）と、その入力に応じて出力されるべき戻り値との組からなるテストパターンを作成する。メソッドに処理を実行させたときに、テストパターンと同じ入出力が行われるか否かにより、正しく動作しているかどうかを検証できる。

【０００４】

ところが、ソフトウェア開発では、テストに必要な工数が非常に大きい。特に品質を重視する場合、プログラミングープログラムテスト工程においてはプログラミング２：テスト８の割合を占めることもある。

【０００５】

そこで、ソフトウェア開発における大きな課題であるテストの効率化のために、テスト支援ツールが開発されている。テスト支援ツールとしては、ドライバ、スタブおよびデバッガ制御コマンド列を自動生成するものがある（例えば、特許文献１参照）。

【０００６】

ここで、ドライバは、被試験モジュールを呼び出すモジュールである。スタブは、被試験モジュールの下位モジュールの動作を擬似するモジュールである。デバッガ制御コマンドは、被試験モジュールの実行に必要な変数の入力データの設定個所の特定、および実行文中で実際に使用されるフィールドの特定を行い、入力データの設定、結果の確認を促すものである。

【０００７】

【特許文献１】

特開平６－２５０８８４号公報

【０００８】

【発明が解決しようとする課題】

しかし、テスト支援ツールで生成可能なテストパターンは、テストに必要な最低限のものである。そのため、信頼性の高いテストを行うには、ソフトウェアの

機能や処理対象として想定される情報に応じて、パラメタと戻り値との様々な組み合わせのテストパターンを追加で用意する必要がある。追加で用意するテストパターンは、ソフトウェアの機能に合わせて、作業者が適宜手入力で設定しなければならない。

【０００９】

しかも、信頼性の高いプログラムテストを行うと、大量（たとえば、平均して１０件／ＫＳｔｅｐ（１０００ステップ中の障害発生件数））の障害が検出される。そして、各障害の内容に応じて、障害修正や仕様変更が行われる。そのため、プログラムの変更内容に合わせたテストパターンの変更作業も頻繁に行われる。

【００１０】

このように、従来は、開発中のソフトウェアのプログラムに変更が加えられる度に煩雑なテストパターン変更作業を行っているため、テストパターンの作成に多大な労力が費やされていた。また、テスト支援ツールを利用してもテストパターンの作成負荷が軽減されないということが、テスト支援ツールの適用阻害要因ともなっている。

【００１１】

本発明はこのような点に鑑みてなされたものであり、テスト対象のプログラムの内容が更新された場合に、更新後のプログラム用のテストパターンを効率的に作成できるテスト支援プログラムおよびテスト支援方法を提供することを目的とする。

【００１２】

【課題を解決するための手段】

本発明では上記課題を解決するために、図１に示すような処理をコンピュータに実行させるテスト支援プログラムが提供される。本発明に係るテスト支援プログラムは、開発対象プログラムで定義された機能に対して入力するパラメタと機能からの戻り値とを含むテストパターンを用いて、開発対象プログラムの動作テストを実施するためのものである。このテスト支援プログラムに従って、コンピュータが以下の処理を実行する。

【００１３】

コンピュータは、開発対象プログラムの内容が更新されたとき、更新後の開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述１を取得する（ステップＳ１）。次に、コンピュータは、更新前の開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述２から、更新後動作記述と共通性の高い更新前動作記述２ａを選択する（ステップＳ２）。さらに、コンピュータは、更新前の開発対象プログラムの動作テストのために作成された複数の更新前テストパターン３から、選択された更新前動作記述２ａの動作テストのための更新前テストパターン３ａを抽出する（ステップＳ３）。そして、コンピュータは、抽出した更新前テストパターン３ａの少なくとも一部を引き継いで、更新後動作記述の動作テストのための更新後テストパターン４を生成する（ステップＳ４）。

【００１４】

これによりに、更新後動作記述１と共通性の高い更新前動作記述２ａに対して設定された更新前テストパターン３ａの内容を引き継いで、更新後動作記述１に対する更新後テストパターン４が生成される。

【００１５】

また、本発明では、上記課題を解決するために、開発対象プログラムで定義された機能に対して入力するパラメタと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するためのテスト支援方法において、前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述１を取得し（ステップＳ１）、更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述２から、前記更新後動作記述と共通性の高い前記更新前動作記述２ａを選択し（ステップＳ２）、更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターン３から、選択された前記更新前動作記述の動作テストのための前記更新前テストパターン３ａを抽出し（ステップＳ３）、抽出した前記更新前テストパターン３ａの少なくとも一部を引き継いで、前記更新後動作記述１の動作テストのための更新後テ

ストパターン４を生成する（ステップＳ４）、ことを特徴とするテスト支援方法が提供される。

【００１６】

これにより、更新後動作記述１と共通性の高い更新前動作記述２ａに対して設定された更新前テストパターン３ａの内容を引き継いで、更新後動作記述１に対する更新後テストパターン４が生成される。

【００１７】

【発明の実施の形態】

以下、本発明の実施の形態を図面を参照して説明する。

まず、実施の形態に適用される発明の概要について説明し、その後、実施の形態の具体的な内容を説明する。

【００１８】

図１は、実施の形態に適用される発明の概念図である。本発明では、開発対象プログラムで定義された機能に対して入力するパラメタと機能からの戻り値とを含むテストパターンを用いて開発対象プログラムの動作テストを実施する。そのために、まず、開発対象プログラムの内容が更新されたとき、更新後の開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述１を取得する（ステップＳ１）。更新後動作記述１は、たとえば、更新後の開発対象プログラムの内容を解析することで取得することができる。なお、更新後の開発対象プログラムに複数の機能が含まれていれば、機能毎の複数の更新後動作記述が生成されることとなる。

【００１９】

次に、更新前の開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述２から、更新後動作記述と共通性の高い更新前動作記述２ａを選択する（ステップＳ２）。なお、複数の更新後動作記述２は、たとえば、更新前の開発対象プログラムの内容を解析することで取得された機能毎の動作記述である。

【００２０】

さらに、更新前の開発対象プログラムの動作テストのために作成された複数の

更新前テストパターン３から、ステップＳ２で選択された更新前動作記述の動作テストのための更新前テストパターン３ａを抽出する（ステップＳ３）。

【００２１】

そして、抽出した更新前テストパターン３ａの少なくとも一部を引き継いで、更新後動作記述１の動作テストのための更新後テストパターン４を生成する（ステップＳ４）。

【００２２】

これにより、更新後動作記述１と共通性の高い更新前動作記述２ａに対して設定された更新前テストパターン３ａの内容を引き継いで、更新後動作記述１に対する更新後テストパターン４が生成される。その結果、開発対象プログラムが更新されたときにもテストパターンを一から作り直す必要が無くなり、テストパターンの作成が容易となる。テストパターンを容易に作成できることで、テスト支援プログラムを用いた動作テストを効率よく行うことができる。

【００２３】

以下、本発明の実施の形態を具体的に説明する。なお、以下の実施の形態では、オブジェクト指向言語のクラスにおける機能（メソッド）に着目して、オブジェクト指向で作成されたソフトウェアのテストを実行するテスト支援装置の例を示す。本発明をオブジェクト指向によるソフトウェア開発に適用することで、オブジェクト指向言語で書かれたソフトウェアプログラムの生産性を向上させることができる。

【００２４】

オブジェクト指向言語でソフトウェア開発を行う場合、テスト対象のクラスの機能が変更されたとき、設計済みのテストパターンを、変更後のメソッドに対応するテストパターンに引き継がせる。オブジェクト指向言語としては、主にｊａｖａ（登録商標）やＣ＋＋などがあるが、以下の実施の形態では、Ｊａｖａ（登録商標）言語のテスト支援装置の例を示す。

【００２５】

オブジェクト指向言語の場合、機能（メソッド）はメソッドパラメタ値の入力パターンにより異なる動作をする。テスト支援ツールではこのメソッドを呼び出

す際のパラメタ値を様々な入力テストパターンとして設計する。その入力テストパターンを用いてメソッドを呼び出すことで、テストを実行できる。

【００２６】

ところで、オブジェクト指向言語によるソフトウェア開発では、クラスの仕様の変更、障害対応による変更などでメソッドのインタフェースの仕様が変更になることが開発中には多々ある。その場合、定義済みのテストパターン（メソッドの呼び出しパラメタの値）が無駄になりテストの効率が悪くなってしまう。そこで、そのような効率の悪化を防ぐために、クラスの仕様変更時にもメソッドにおけるテストパターンを自動で引き継ぐ機構を、以下に示すテスト支援装置によって実現する。

【００２７】

なお、以下の実施の形態では、機能（メソッド）のインタフェースの属性に着目し、一定のルールで新機能が旧機能に変更されたかを自動判断しテストパターンを引継ぐ機構と、判断できない場合には引継ぎ可能なテストパターンを利用者に提示する機構とを提供する。着目するインタフェースとしては「メソッド名（機能の名前）」、「メソッドパラメタ（機能の動作条件）」、「メソッド戻り値（機能の実行結果）」などが挙げられる。着目した属性には優先順位が付けられ、一致した属性の優先順位に基づいたルールにより、変更前メソッドと変更後メソッドの対応付けが自動で行われ、テストパターンの自動引継ぎが行われる。

【００２８】

まず、テスト支援装置のハードウェア構成について説明する。

図２は、本発明の実施の形態に用いるテスト支援装置のハードウェア構成例を示す図である。テスト支援装置１００は、ＣＰＵ(Central Processing Unit)１０１によって装置全体が制御されている。ＣＰＵ１０１には、バス１０７を介してＲＡＭ(Random Access Memory)１０２、ハードディスクドライブ（ＨＤＤ:Hard Disk Drive）１０３、グラフィック処理装置１０４、入力インタフェース１０５、および通信インタフェース１０６が接続されている。

【００２９】

ＲＡＭ１０２には、ＣＰＵ１０１に実行させるＯＳ(Operating System)のプロ

グラムやアプリケーションプログラムの少なくとも一部が一時的に格納される。また、ＲＡＭ１０２には、ＣＰＵ１０１による処理に必要な各種データが格納される。ＨＤＤ１０３には、ＯＳやアプリケーションプログラムが格納される。

【００３０】

グラフィック処理装置１０４には、モニタ１１が接続されている。グラフィック処理装置１０４は、ＣＰＵ１０１からの命令に従って、画像をモニタ１１の画面に表示させる。入力インタフェース１０５には、キーボード１２とマウス１３とが接続されている。入力インタフェース１０５は、キーボード１２やマウス１３から送られてくる信号を、バス１０７を介してＣＰＵ１０１に送信する。

【００３１】

通信インタフェース１０６は、ネットワーク１０に接続されている。通信インタフェース１０６は、ネットワーク１０を介して、他のテスト支援装置との間でデータの送受信を行う。

【００３２】

以上のようなハードウェア構成によって、本実施の形態のテスト支援装置の機能を実現することができる。

図３は、テスト支援装置の有する処理機能を示すブロック図である。図３に示すように、テスト支援装置１００は、テストクラス記憶部１１０、構文・意味解析部１２０、第１の解析結果記憶部１３０、テストパターン作成部１４０、テストパターン記憶部１５０、第２の解析結果記憶部１６０、更新時自動引継ぎ部１７０、テスト実行部１８０および手動引継ぎ部１９０を有している。またテスト支援装置１００には、モニタ１１やキーボード１２等の入出力装置１４が接続されている。

【００３３】

テストクラス記憶部１１０は、テストの対象となるクラス（テストクラス）を記憶する。クラスは、共通の属性(特性)を有するインスタンス（オブジェクト）の集合である。クラスは、個々のインスタンスを定義するひな形と考えることができる。クラスには、そのクラスに属する各インスタンスが共通にもつ属性項目の集合と手続きの集合がまとめて記述される。

【００３４】

構文・意味解析部１２０は、テストクラスの構文やその意味を解析し、解析結果情報を生成する。構文・意味解析部１２０は、パーサとも呼ばれる。構文・意味解析部１２０は、新規に作成されたテストクラスの解析結果情報を、第１の解析結果記憶部１３０に格納する。また、構文・意味解析部１２０は、内容が更新されたテストクラスの解析結果情報を、第２の解析結果記憶部１６０に格納する。

【００３５】

第１の解析結果記憶部１３０は、新規に作成されたテストクラスの解析結果を記憶する。

テストパターン作成部１４０は、第１の解析結果記憶部１３０に格納された解析結果情報を基にテストパターンを作成する。テストパターン作成部１４０は、作成したテストパターンをテストパターン記憶部１５０に格納する。

【００３６】

テストパターン記憶部１５０は、テストパターン作成部１４０で作成されたテストパターン、および更新時自動引継ぎ部１７０や更新時手動引継ぎ部１９０で引き継がれたテストパターンを記憶する。

【００３７】

第２の解析結果記憶部１６０は、既存のテストクラスから変更されたテストクラスの解析結果を記憶する。

更新時自動引継ぎ部１７０は、テストパターン記憶部１５０に格納されたテストパターンを基に、第２の解析結果記憶部１６０内の解析結果情報に応じたテストパターンを作成する。更新時自動引継ぎ部１７０は、作成したテストパターンを、テストパターン記憶部１５０に格納する。

【００３８】

テスト実施部１８０は、テストパターン記憶部１５０に格納されているテストパターンに基づいて、テストクラス記憶部１１０に格納されているテストクラスの動作テストを行う。

【００３９】

具体的には、テスト実施部１８０は、テストドライバ生成部１８１、テストスタブ生成部１８２、自動実行部１８３、および結果検証部１８４を有している。テストドライバ生成部１８１は、テストドライバを生成する。テストドライバは、被試験モジュールを呼び出すプログラムモジュールである。テストスタブ生成部１８２は、テストスタブを生成する。テストスタブは、試験対象のテストクラスの下位構造のクラスの動作を擬似的に実現するプログラムである。自動実行部１８３は、テストドライバ、テストスタブ、テストパターン等を用いて、テストクラスのテストを実行する。

【００４０】

手動引継ぎ部１９０は、新規に作成されたテストクラスのテストパターンから、そのテストクラスの更新後のテストクラスのテストパターンへ、操作入力に応答してテストパターンを引き継がせる。

【００４１】

以上のような構成によって、テストパターンの自動引き継ぎが可能となる。

図４は、テストパターン自動引き継ぎの入出力関係を示す図である。更新前のテストクラス１１１は、構文・意味解析部１２０に入力される。すると、構文・意味解析部１２０により構文・意味の解析が行われ、更新前の解析結果情報１３１が生成される。生成された解析結果情報１３１は、テストパターン作成部１４０に入力される。テストパターン作成部１４０は、入力された解析結果情報１３１に基づいて、更新前のテストパターン１５１を作成する。テストパターン１５１は、テスト実施部１８０によって参照され、そのテストパターン１５１に基づいて更新前のテストクラス１１１の動作検証が行われる。

【００４２】

更新前のテストクラス１１１に関する動作検証で、不具合が見付かった場合には、テストクラス１１１の内容が更新され、更新後のテストクラス１１２が生成される。更新後のテストクラス１１２は、構文・意味解析部１２０に入力される。すると、構文・意味解析部１２０により構文・意味の解析が行われ、更新後の解析結果情報１６１が生成される。

【００４３】

その後、更新前の解析結果情報１３１と更新後の解析結果情報１６１とが更新時自動引継ぎ部１７０に入力される。更新時自動引継ぎ部１７０は、テストクラスの更新前後におけるそれぞれの解析結果情報１３１，１６１を比較する。そして、更新時自動引継ぎ部１７０は、変更部分の引継ルールに従い引き継ぐことができるテストパターンを決定する。次に、更新時自動引継ぎ部１７０は、更新前のテストパターン１５１から、更新後のテストクラスに対する動作テストに引き継ぐことができるテストパターンを抽出する。そして、更新時自動引継ぎ部１７０は、抽出したテストパターンを、更新後のテストパターン１５２に設定する。

【００４４】

更新時自動引継ぎ部１７０によるテストパターンの自動的な引き継ぎが行われた後、手動引継ぎ部１９０によって、更新前のテストパターン１５１と更新後のテストパターン１５２との内容が、入出力装置１４の画面に比較表示される。このとき、自動引継ができなかった部分について、強調表示される。そして、作業者がキーボード１２やマウス１３を操作して、テストパターンの引き継ぎ指示を入力すると、その指示に従って手動引継ぎ部１９０が更新前のテストパターン１５１内のテストパターンを、更新後のテストパターン１５２に設定する。このようにして生成された更新後のテストパターン１５２に基づいて、テスト実施部１８０による動作検証が行われる。

【００４５】

ところで、テストクラス１１１，１１２は、様々なメソッド（機能）を有している。メソッドは、テストクラス１１１，１１２を記述したソースプログラム内の一部の動作記述によって、その動作が定義されている。メソッドは、複数の情報が含まれており、各情報は属性を有している。

【００４６】

図５は、メソッドに含まれる情報の属性を示す図である。これはｊａｖａ（登録商標）言語定義でのメソッド２０の構造を表している。図５に示すように、メソッド２０は、修飾子２１、メソッド戻り値２２、メソッド名２３、およびメソッドパラメタ２４の属性の情報を有している。

【００４７】

図６は、解析結果情報の保存形式例を示す図である。解析結果情報１３１は、複数のメソッド情報テーブル１３１ａと複数のメソッドパラメタテーブル１３１ｂとで構成されている。メソッド情報テーブル１３１ａには、メソッドＩＤ、クラス名、メソッド名、メソッド戻り値、パラメタ数などの情報が設定される。メソッドパラメタテーブル１３１ｂには、パラメタＩＤ、メソッドＩＤ、パラメタ名、パラメタ型などの情報が設定される。

【００４８】

メソッド情報テーブル１３１ａとメソッドパラメタテーブル１３１ｂとは、メソッドＩＤによって関係づけられている。互いに関係するメソッド情報テーブル１３１ａに登録されたメソッド情報とメソッドパラメタテーブル１３１ｂに登録されたメソッドパラメタとは、１対Ｎ（Ｎは０以上の整数）の関係で存在している。すなわち、１つのメソッド情報に対して、１以上のメソッドパラメタが関係づけられる。

【００４９】

以下、Employeeクラスをテストクラスとして、動作テストを行う場合の例について具体的に説明する。

図７は、更新前のテストクラス例を示す図である。この例では、テストクラス１１１において、Ｊａｖａ（登録商標）言語によるメソッドが記述されている。

【００５０】

第１のメソッド１１１ａは、従業員名からの従業員検索機能を定義したメソッドである。第２のメソッド１１１ｂは、従業員氏名、年齢からの従業員検索機能を定義したメソッドである。第３のメソッド１１１ｃは、従業員氏名、年齢、部門コードからの従業員検索機を定義したメソッドである。第４のメソッド１１１ｄは、従業員氏名、部門コードからの従業員検索機能を定義したメソッドである。第５のメソッド１１１ｅは、従業員コードからの従業員情報の更新機能を定義したメソッドである。第６のメソッド１１１ｆは、従業員コードからの従業員情報の更新機能を定義したメソッドである。

【００５１】

このようなテストクラス１１１が構文・意味解析部１２０で解析され、解析結

果情報が生成される。以下、EmployeeクラスのメソッドsearchEmployeebyNoの解析結果情報１３１の保存例を示す。図４で示したように、解析結果情報は、メソッド情報テーブルとメソッドパラメタテーブルとで構成される。

【００５２】

図８は、更新前のメソッド情報テーブルの例を示す図である。メソッド情報テーブル１３１ａの各欄の横方向に並べられた情報同士が互いに関連づけられ、関連づけられた情報が１つのメソッド情報を構成している。

【００５３】

メソッドＩＤの欄に、第１のメソッド１１１ａ～第６のメソッド１１ｆに対応する各メソッド情報に対して順番に１～６の番号が設定されている。クラス名の欄には、全てのメソッド情報に対してEmployeeと設定されている。メソッド名の欄には、各メソッド情報のメソッド名が設定されている。メソッドＩＤ「１」～「４」のメソッド名は「searchEmployee」である。メソッドＩＤ「５」のメソッド名は「updateEmployee」である。メソッドＩＤ「６」のメソッド名は「deleteEmployee」である。戻り値型の欄には、メソッドの戻り値の型が設定されている。メソッドＩＤ「１」～「４」の戻り値の型は「EmployeeData」である。メソッドＩＤ「５」の戻り値の型は「boolean」である。メソッドＩＤ「６」の戻り値の型は「int」である。

【００５４】

パラメタ数の欄には、関連づけられたメソッドパラメタの数が設定されている。メソッドＩＤ「１」の第１のメソッド１１１ａには、１つのメソッドパラメタが関連づけられている。メソッドＩＤ「２」の第２のメソッド１１１ｂには、２つのメソッドパラメタが関連づけられている。メソッドＩＤ「３」の第３のメソッド１１１ｃには、３つのメソッドパラメタが関連づけられている。メソッドＩＤ「４」の第４のメソッド１１１ｄには、２つのメソッドパラメタが関連づけられている。メソッドＩＤ「５」の第５のメソッド１１１ｅには、３つのメソッドパラメタが関連づけられている。メソッドＩＤ「６」の第６のメソッド１１１ｆには、１つのメソッドパラメタが関連づけられている。

【００５５】

図９は、更新前のメソッドパラメタテーブルの例を示す図である。メソッドパラメタテーブル１３１ｂの各欄の横方向に並べられた情報同士が互いに関連づけられ、関連づけられた情報が１つのメソッドパラメタを構成している。

【００５６】

パラメタＩＤの欄には、各メソッドパラメタに対して順番に１～１２の番号が設定されている。

メソッドＩＤの欄には、各メソッドパラメタが関連づけられているメソッド情報のメソッドＩＤが設定されている。パラメタＩＤ「１」のメソッドパラメタは、メソッドＩＤ「１」のメソッド情報に関連づけられている。パラメタＩＤ「２」、「３」のメソッドパラメタは、メソッドＩＤ「２」のメソッド情報に関連づけられている。パラメタＩＤ「４」～「６」のメソッドパラメタは、メソッドＩＤ「３」のメソッド情報に関連づけられている。パラメタＩＤ「７」、「８」のメソッドパラメタは、メソッドＩＤ「４」のメソッド情報に関連づけられている。パラメタＩＤ「９」～「１１」のメソッドパラメタは、メソッドＩＤ「５」のメソッド情報に関連づけられている。パラメタＩＤ「１２」のメソッドパラメタは、メソッドＩＤ「６」のメソッド情報に関連づけられている。

【００５７】

パラメタ名の欄には、各メソッドパラメタの名称が設定されている。パラメタＩＤ「１」、「２」、「４」、「７」、「９」のパラメタ名は「EmployeeName」である。パラメタＩＤ「３」、「５」、「１０」のパラメタ名は「age」である。パラメタＩＤ「６」、「８」「１１」のパラメタ名は「code」である。パラメタＩＤ「１２」のパラメタ名は「EmployeeNo」である。

【００５８】

パラメタ型の欄には、各メソッドパラメタのデータ型が設定されている。メソッドＩＤ「１」、「２」、「４」、「７」、「９」のパラメタ型は「String」である。メソッドＩＤ「３」、「５」、「６」、「８」、「１０」～「１２」のパラメタ型は「int」である。

【００５９】

このような解析結果情報に基づいて、テストパターン作成部１４０によりテス

トパターンが作成される。なお、テストパターン作成部１４０は、テストパターンの全てを自動で作成するのではなく、作業者からの操作入力に基づいたテストパターンの生成も行う。例えば、テストパターン作成部１４０は、メソッド毎に、正常な戻り値が得られるテストパターンと正常な戻り値が得られないテストパターンとを生成する。そして、テストパターン作成部１４０は、入出力装置１４による作業者からの操作入力に応答して、様々なバリエーションのテストパターンを生成する。このように、作業者が任意に入力したテストパターンを用いれば、信頼性の高い動作テストが可能である。

【００６０】

図１０～図１４に、メソッド毎のテストパターンの例を示している。なお、以下の説明において、メソッド毎のテストパターンの集合をテストパターン群と呼ぶこととする。

【００６１】

図１０は、テストクラス変更前の第１のメソッドのテストパターンの例を示す図である。第１のメソッド１１１ａのテストパターン群１５１ａでは、各テストパターンに対して、識別番号（Ｎｏ．）、メソッド名、パラメタ、正常／異常、戻り値、およびテストの観点の欄が設けられている。各欄の横方向に並べられた情報同士が互いに関連づけられ、テストパターンを構成している。

【００６２】

識別番号の欄には、テストパターンの識別番号が設定されている。第１のメソッド１１１ａに対するテストパターンのグループにグループ番号Ｍ１が設定され、個々のテストパターンの識別番号は、グループ番号Ｍ１と枝番とで表される。図１０の例では、３つのテストパターンに「Ｍ１－１」～「Ｍ１－３」の識別番号が設定されている。

【００６３】

メソッド名の欄には、テストパターンがテストの対象としているメソッドの名称が設定される。図１０の例は第１のメソッド１１１ａに対するテストパターンであるため、全てのテストパターンに対して第１のメソッド１１１ａの「searchEmployee」のパラメタが設定されている。このパラメタは、従業者の氏名を表す

パラメタである。

【００６４】

パラメタの欄には、第１のメソッド１１１ａに対して、パラメタ「EmployeeName」として入力する値が設定されている。識別番号「Ｍ１－１」のテストパターンでは、「Ｆ通　太郎」という値が入力される。識別番号「Ｍ１－２」のテストパターンでは、空白の値が入力される。識別番号「Ｍ１－３」のテストパターンでは、Null（空を示す情報）が入力される。

【００６５】

正常／異常の欄には、入力するパラメタが正常な値なのか、異常な値なのかを示す情報が設定される。識別番号「Ｍ１－１」のテストパターンで入力されるパラメタは正常な値である。識別番号「Ｍ１－２」のテストパターンで入力されるパラメタは異常な値である。識別番号「Ｍ１－３」のテストパターンで入力されるパラメタは異常な値である。

【００６６】

戻り値の欄には、第１のメソッド１１１ａからの戻り値「EmployeeData」の種類を示す情報が設定される。識別番号「Ｍ１－１」のテストパターンを入力したときの戻り値は、検索結果である。識別番号「Ｍ１－２」のテストパターンを入力したときの戻り値は、Nullである。識別番号「Ｍ１－３」のテストパターンを入力したときの戻り値は、Nullである。

【００６７】

テストの観点の欄は、テストパターンによって実行されるテストの目的が示されている。たとえば、識別番号「Ｍ１－１」のテストパターンは正常系のテストである。識別番号「Ｍ１－２」のテストパターンは、氏名が空白の場合の異常系のテストである。識別番号「Ｍ１－３」のテストパターンは、氏名がnullの場合の異常系のテストである。なお、テストの観点の欄は、テストパターンの意味を説明するために設けられた項目であり、テストパターンの自動引継やテスト実行のために必須の情報ではない。

【００６８】

他のメソッドに対するテストパターンについては、第１のメソッド１１１ａの

テストパターンと異なる部分についてのみ説明する。

図１１は、テストクラス変更前の第２のメソッドのテストパターンの例を示す図である。第２のメソッド１１１ｂのテストパターン群１５１ｂでは、グループ番号Ｍ２が設定されている。図１２の例では、「Ｍ２－１」～「Ｍ２－７」の７つのテストパターンが設定されている。メソッド１１１ｂのメソッド名は、「searchEmployee」である。

【００６９】

メソッド１１１ｂのテストパターンでは、「EmployeeName」と「Age」とのパラメタが設定される。「Age」は従業者の年齢を示すパラメタである。

この例では、正常系のテストパターン（識別番号「Ｍ２－１」）や従業者の氏名に関する異常系のテストパターン（識別番号「Ｍ２－２」、識別番号「Ｍ２－３」）に加え、従業者の年齢に関する異常系のテストパターンが設定されている。たとえば、年齢が０の場合の異常系（識別番号「Ｍ２－４」）、年齢が従業員雇用範囲外の場合（小さすぎと大きすぎ）の異常系（識別番号「Ｍ２－５」、識別番号「Ｍ２－６」）、年齢がnullの場合の異常系（識別番号「Ｍ２－７」）である。

【００７０】

図１２は、テストクラス変更前の第３のメソッドのテストパターンの例を示す図である。第３のメソッド１１１ｃのテストパターン群１５１ｃでは、グループ番号Ｍ３が設定されている。図１２の例では、「Ｍ３－１」～「Ｍ３－１０」の１０個のテストパターンが設定されている。メソッド１１１ｃのメソッド名は、「searchEmployee」である。

【００７１】

メソッド１１１ｃのテストパターンでは、「EmployeeName」、「Age」、および「code」のパラメタが設定される。「code」は各従業員を一意に識別するための従業員コードを示すパラメタである。

【００７２】

この例では、正常系のテストパターン（識別番号「Ｍ３－１」）や従業者の氏名や年齢に関する異常系のテストパターン（識別番号「Ｍ３－２」～識別番号「

Ｍ３－７」）に加え、従業者の従業員コードに関する異常系のテストパターンが設定されている。たとえば、従業員コードが０の場合の異常系（識別番号「Ｍ３－８」）、従業員コードがNullの場合の異常系（識別番号「Ｍ３－９」）である。なお、この例では、全てのパラメタがnullの場合の異常系（識別番号「Ｍ３－１０」）に関するテストパターンも設定されている。

【００７３】

図１３は、テストクラス変更前の第４のメソッドのテストパターンの例を示す図である。第４のメソッド１１１ｄのテストパターン群１５１ｄでは、グループ番号Ｍ４が設定されている。図１３の例では、「Ｍ４－１」～「Ｍ４－６」の６個のテストパターンが設定されている。メソッド１１１ｄのメソッド名は、「searchEmployee」である。メソッド１１１ｄのテストパターンでは、「EmployeeName」と「code」とのパラメタが設定される。

【００７４】

この例では、正常系のテストパターン（識別番号「Ｍ４－１」）や従業者の氏名や従業員コードに関する異常系のテストパターン（識別番号「Ｍ４－２」～識別番号「Ｍ４－６」）が設定されている。

【００７５】

図１４は、テストクラス変更前の第５のメソッドのテストパターンの例を示す図である。第５のメソッド１１１ｅのテストパターン群１５１ｅでは、グループ番号Ｍ５が設定されている。図１４の例では、「Ｍ５－１」～「Ｍ５－１０」の１０個のテストパターンが設定されている。メソッド１１１ｅのメソッド名は、「UpdateEmployee」である。メソッド１１１ｅのテストパターンでは、「EmployeeName」、「Age」、および「code」のパラメタが設定される。

【００７６】

この例では、図１２に示した第３のメソッド１１１ｃのテストパターンと同様に、正常系のテストパターン（識別番号「Ｍ５－１」）や従業者の氏名、年齢、従業員コードに関する異常系のテストパターン（識別番号「Ｍ５－２」～識別番号「Ｍ５－１０」）が設定されている。

【００７７】

図１５は、テストクラス変更前の第６のメソッドのテストパターンの例を示す図である。第６のメソッド１１１ｆのテストパターン群１５１ｆでは、グループ番号Ｍ６が設定されている。図１５の例では、「Ｍ６－１」～「Ｍ６－４」の４個のテストパターンが設定されている。メソッド１１１ｆのメソッド名は、「deleteEmployee」である。メソッド１１１ｆのテストパターンでは、「code」のパラメタが設定される。

【００７８】

この例では、正常系のテストパターン（識別番号「Ｍ６－１」）や従業員コードに関する異常系のテストパターン（識別番号「Ｍ６－２」～識別番号「Ｍ６－４」）が設定されている。従業員コードの異常系には、従業員コードが０の場合、従業員コードがnullの場合、従業員コードが該当なしの場合がある。なお、正常系の戻り値は「True」であり、異常系の戻り値は「False」である。

【００７９】

このようなテストパターンを用いて、テストクラスが設計通りに動作するかどうかをテストすることができる。

なお、テストパターン作成部１４０で自動作成できないテストパターンとして、たとえば、第２のメソッド１１１ｂの識別番号「Ｍ２－５」や識別番号「Ｍ２－６」のテストパターン、第６のメソッド１１１ｆの識別番号「Ｍ６－４」のテストパターンなどがある。第２のメソッド１１１ｂの識別番号「Ｍ２－５」や識別番号「Ｍ２－６」のテストパターンは、従業員雇用範囲である年齢層が分かっていないと設定できない。また、第６のメソッド１１１ｆの識別番号「Ｍ６－４」のテストパターンは、該当者の以存在しない従業員コードが分かっていないと設定できない。そのため、これらのテストパターンについては、テストクラスを運用するときの環境に合わせたパラメタを作業者が入力する必要がある。

【００８０】

このようなテストの設定情報（テストパターンの集合）を機能（メソッド）が変更になった場合も引き継いで利用できれば、テストクラスの運用環境に合わせて作業者が設定したテストクラスの再設定が不要となる。そこで、本実施の形態では、機能の変更パターンとその場合の引継ぎルールを決めることにより上記設

定情報を新しい機能（メソッド）でも利用可能とする。

【００８１】

ここで、引き継ぎのルールとして、機能（メソッド）の属性に着目する。上記の例では、３つの属性がある。機能の実現に不可欠な属性であるこの３属性に優先順位をつける。そして、更新前のメソッドと更新後のメソッドとの間で、属性の一致／不一致によって、テストパターンの引き継ぎの可否を決定する。更新前のメソッドといずれかの属性が一致する更新後のメソッドが複数有れば、高優先度の属性が一致する更新後のメソッドにテストパターンを引き継がせる。

【００８２】

本実施の形態における各属性の優先順位は、以下の通りである。

第１優先　メソッド名（機能を表す表現）

第２優先　メソッドパラメタ（機能が動作するときの条件）

第３優先　メソッド戻り値（機能の結果）

ここで、各属性を引き継ぎ可否の判定基準に採用した理由について説明する。

【００８３】

・第１の優先をメソッド名とした理由

オブジェクト指向プログラミングでは、異なるメソッドに同じメソッド名が与えられることない。また、一般的に、メソッドの内容が部分的に変更されてもメソッド名が変更されることはない。そのため、更新前のテストクラス１１１と更新後のテストクラス１１２とにおいて、メソッド名が同一のメソッドであれば、そのメソッドのテストパターンにおけるメソッドパラメタと戻り値とを引き継げる。したがって、本実施の形態では、メソッド名を第１優先の属性としている。

【００８４】

・第２の優先をメソッドパラメタとした理由

テストクラスに対するテストでは、メソッドに対して正常なパラメタが入力されたときに正しい戻り値が返され、メソッドに対して異常なパラメタが入力されたときにエラー等の異常を表す戻り値が返されることが検証される。テストパターンとしては、正常なパラメタの組を含むものと、少なくとも一部に異常なパタメタを含むパタメタの組を含むものとが用意される。メソッドを更新する場合、

入力されるパラメタが同じままで、動作内容に変更が加えられる場合も多い。そのため、更新前のテストクラス１１１と更新後のテストクラス１１２とにおいて、入力されるメソッドパラメタの組が同じであれば、そのメソッドのテストパターンのメソッドパラメタを引き継げる。したがって、本実施の形態では、メソッドパラメタ名を第２優先の属性としている。

【００８５】

・第３の優先をメソッド戻り値とした理由

更新前のテストクラス１１１と更新後のテストクラス１１２とにおいて、メソッドパラメタが異なっていても、メソッド戻り値が同じであれば、メソッド戻り値を引き継げる。そこで、本実施の形態では、メソッド戻り値を第３優先の属性としている。

【００８６】

以下、テストクラス更新時のテストパターンの引き継ぎ状況について説明する。

図１６は、更新後のテストパターンの例を示す図である。図７に示した更新前のテストパターンからの変更部分は以下の通りである。

【００８７】

第１のメソッド１１２ａは、修飾子が変更されている。更新前の第１のメソッド１１１ａにおいて「public」であった修飾子が、更新後の第１のメソッド１１２ａでは「private」となっている。

【００８８】

第２のメソッド１１２ｂは、メソッド戻り値が変更されている。更新前の第２のメソッド１１１ｂにおいて「EmployeeData」であったメソッド戻り値が、更新後の第２のメソッド１１２ｂでは「EmployeeData[]」となっている。

【００８９】

第３のメソッド１１２ｃは、メソッドパラメタが変更されている。更新前の第３のメソッド１１１ｃにおいて「String EmployeeName,int Age,int code」であったメソッドパラメタが、更新後の第３のメソッド１１２ｃでは「String EmployeeName,long Age,int section,Boolean retire」となっている。

【００９０】

第４のメソッド１１２ｄは、メソッド戻り値とメソッドパラメタとが変更されている。更新前の第４のメソッド１１１ｄにおいて「EmployeeData」であったメソッド戻り値が、更新後の第４のメソッド１１２ｄでは「EmployeeData[]」となっている。更新前の第３のメソッド１１１ｄにおいて「String EmployeeName,int code」であったメソッドパラメタが、更新後の第４のメソッド１１２ｄでは「String EmployeeName,int section」となっている。

【００９１】

第５のメソッド１１２ｅは、メソッド名が変更されている。更新前の第５のメソッド１１１ｅにおいて「updateEmployee」であったメソッド名が、更新後の第５のメソッド１１２ｅでは「updateEmployeebyNo」となっている。

【００９２】

第６のメソッド１１２ｆは、メソッド名とメソッドパラメタとが変更されている。更新前の第６のメソッド１１１ｆにおいて「deleteEmployee」であったメソッド名が、更新後の第６のメソッド１１２ｆでは「deleteEmployeebyNo」となっている。更新前の第６のメソッド１１１ｆにおいて「int EmployeeNo」であったメソッドパラメタが、更新後の第６のメソッド１１２ｆでは「int EmployeeNo,StringEmployeeName」となっている。

【００９３】

このようなメソッド１１２ａ～１１２ｆを含む更新後のテストクラス１１２が生成されると、構文・意味解析部１２０において解析が行われ、更新後の解析結果情報１６１が生成される。

【００９４】

図１７は、更新後のメソッド情報テーブルの例を示す図である。メソッド情報テーブル１６１ａの各欄の横方向に並べられた情報同士が互いに関連づけられ、関連づけられた情報が１つのメソッド情報を構成している。

【００９５】

メソッドＩＤの欄に、各メソッド情報に対して順番に１～６の番号が設定されている。クラス名の欄には、全てのメソッド情報に対してEmployeeと設定されて

いる。

【００９６】

メソッド名の欄には、各メソッド情報のメソッド名が設定されている。メソッドＩＤ「１」～「４」のメソッド名は「searchEmployee」である。メソッドＩＤ「５」のメソッド名は「updateEmployeeNo」である。メソッドＩＤ「６」のメソッド名は「deleteEmployeeNo」である。

【００９７】

戻り値型の欄には、メソッドの戻り値の型が設定されている。メソッドＩＤ「１」、「３」の戻り値の型は「EmployeeData」である。メソッドＩＤ「２」、「４」の戻り値の型は「EmployeeData[]」である。メソッドＩＤ「５」の戻り値の型は「Boolean」である。メソッドＩＤ「６」の戻り値の型は「int」である。

【００９８】

パラメタ数の欄には、関連づけられたメソッドパラメタの数が設定されている。メソッドＩＤ「１」のメソッドには、１つのメソッドパラメタが関連づけられている。メソッドＩＤ「２」のメソッドには、２つのメソッドパラメタが関連づけられている。メソッドＩＤ「３」のメソッドには、４つのメソッドパラメタが関連づけられている。メソッドＩＤ「４」のメソッドには、２つのメソッドパラメタが関連づけられている。メソッドＩＤ「５」のメソッドには、３つのメソッドパラメタが関連づけられている。メソッドＩＤ「６」のメソッドには、１つのメソッドパラメタが関連づけられている。

【００９９】

図１８は、更新後のメソッドパラメタテーブルの例を示す図である。メソッドパラメタテーブル１６１ｂの各欄の横方向に並べられた情報同士が互いに関連づけられ、関連づけられた情報が１つのメソッドパラメタを構成している。

【０１００】

パラメタＩＤの欄には、各メソッドパラメタに対して順番に１～１４の番号が設定されている。

メソッドＩＤの欄には、各メソッドパラメタが関連づけられているメソッド情報のメソッドＩＤが設定されている。パラメタＩＤ「１」のメソッドパラメタは

、メソッドＩＤ「１」のメソッド情報に関連づけられている。パラメタＩＤ「２」、「３」のメソッドパラメタは、メソッドＩＤ「２」のメソッド情報に関連づけられている。パラメタＩＤ「４」～「７」のメソッドパラメタは、メソッドＩＤ「３」のメソッド情報に関連づけられている。パラメタＩＤ「８」、「９」のメソッドパラメタは、メソッドＩＤ「４」のメソッド情報に関連づけられている。パラメタＩＤ「１０」～「１２」のメソッドパラメタは、メソッドＩＤ「５」のメソッド情報に関連づけられている。パラメタＩＤ「１３」、「１４」のメソッドパラメタは、メソッドＩＤ「６」のメソッド情報に関連づけられている。

【０１０１】

パラメタ名の欄には、各メソッドパラメタの名称が設定されている。パラメタＩＤ「１」、「２」、「４」、「８」、「１０」、「１４」のパラメタ名は「EmployeeName」である。パラメタＩＤ「３」、「５」、「１１」のパラメタ名は「Age」である。パラメタＩＤ「６」、「９」のパラメタ名は「Section」である。パラメタＩＤ「７」のパラメタ名は「Retire」である。パラメタＩＤ「１２」のパラメタ名は「Code」である。パラメタＩＤ「１３」のパラメタ名は「EmployeeNo」である。

【０１０２】

パラメタ型の欄には、各メソッドパラメタのデータ型が設定されている。メソッドＩＤ「１」、「２」、「４」、「８」、「１０」、「１４」のパラメタ型は「String」である。メソッドＩＤ「３」、「６」、「９」、「１１」～「１３」のパラメタ型は「int」である。メソッドＩＤ「５」のパラメタ型は「long」である。メソッドＩＤ「７」のパラメタ型は「Boolean」である。

【０１０３】

図１７、図１８に示したような更新後の解析結果情報が生成されると、更新時自動引継ぎ部１７０により、テストパターン自動引継処理が行われる。図１９～図２１に、テストパターン自動引継処理のフローチャートを示す。

【０１０４】

図１９は、テストパターン自動引継処理の基本的なメインのフローチャートである。以下、図１９に示す処理をステップ番号に沿って説明する。なお、以下の

処理は、全て更新時自動引継ぎ部１７０が行う処理である。

【０１０５】

［ステップＳ１１］更新後の解析結果情報１６１から未処理のメソッド情報のレコードを１つ読み込む。具体的には、メソッド情報テーブル１６１ａからメソッド情報を読み込み、読み込んだメソッド情報に関連づけられたメソッドパラメタのレコードをメソッドパラメタテーブル１６１ｂから読み込む。なお、メソッド情報とメソッドパラメタとは、メソッドＩＤによって関連づけられている。すなわち、同じメソッドＩＤを有するメソッド情報とメソッドパラメタとが関連づけられている。

【０１０６】

［ステップＳ１２］処理対象のメソッドが無くなった（メソッド終了）か否かを判断する。具体的には、ステップＳ１１における読み込み処理で、読み込み対象となる未処理のメソッドが発見されなかったときに、メソッド終了と判断される。メソッド終了の場合には、テストパターン自動引継処理が終了する。メソッド終了でない（ステップＳ１１でメソッド情報のレコードが読み込まれた）場合には、処理がステップＳ１３に進められる。

【０１０７】

［ステップＳ１３］ステップＳ１１で読み込んだメソッド情報と３属性の情報（メソッド名、パラメタ名とパラメタ型、および戻り値型）が一致する更新前のメソッド情報が、更新前の解析結果情報１３１内にあるか否かを判定する（第１の判定）。該当するメソッド情報があれば、処理がステップＳ１４に進められる。該当するメソッド情報がなければ、処理がステップＳ１６に進められる。

【０１０８】

［ステップＳ１４］ステップＳ１３で該当すると判断されたメソッド情報に対応するテストパターンにおけるメソッドパラメタの情報を、ステップＳ１１で読み込んだメソッド情報のテストパターンに引き継ぐ。

【０１０９】

［ステップＳ１５］ステップＳ１３で該当すると判断されたメソッド情報に対応するテストパターンにおける戻り値の情報を、ステップＳ１１で読み込んだメ

ソッド情報のテストパターンに引き継ぐ。その後、処置がステップＳ１１に進められる。

【０１１０】

［ステップＳ１６］ステップＳ１１で読み込んだメソッド情報と、戻り値以外の２属性の情報（メソッド名、およびパラメタ名とパラメタ型）が一致する更新前のメソッド情報が、更新前の解析結果情報１３１内にあるか否かを判定する（第２の判定）。該当するメソッド情報があれば、処理がステップＳ１７に進められる。該当するメソッド情報がなければ、処理がステップＳ１９に進められる。

【０１１１】

［ステップＳ１７］ステップＳ１６で該当すると判断されたメソッド情報に対応するテストパターンにおけるメソッドパラメタの情報を、ステップＳ１１で読み込んだメソッド情報のテストパターンに引き継ぐ。

【０１１２】

［ステップＳ１８］メソッド戻り値の情報を自動生成し、ステップＳ１１で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処理がステップＳ１１に進められる。

【０１１３】

［ステップＳ１９］ステップＳ１１で読み込んだメソッド情報と、パラメタ（パラメタ名とパラメタ型）以外の２属性の情報（メソッド名と戻り値型）が一致する更新前のメソッド情報が、更新前の解析結果情報１３１内にあるか否かを判定する（第３の判定）。該当するメソッド情報があれば、処理がステップＳ２０に進められる。該当するメソッド情報がなければ、処理がステップＳ２２に進められる。

【０１１４】

［ステップＳ２０］不一致パラメタ引継処理が行われる。この処理の詳細は後述する。

［ステップＳ２１］ステップＳ１９で該当すると判断されたメソッド情報に対応するテストパターンにおける戻り値の情報を、ステップＳ１１で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処置がステップＳ１１に進め

られる。

【０１１５】

［ステップＳ２２］ステップＳ１１で読み込んだメソッド情報と、メソッド名のみが一致し、他の２属性の情報（パラメタ名とパラメタ型、および戻り値型）が不一致の更新前のメソッド情報が、更新前の解析結果情報１３１内にあるか否かを判定する（第４の判定）。該当するメソッド情報があれば、処理がステップＳ２３に進められる。該当するメソッド情報がなければ、処理がステップＳ２５に進められる。

【０１１６】

［ステップＳ２３］不一致パラメタ引継処理が行われる。この処理の詳細は後述する。

［ステップＳ２４］メソッド戻り値の情報を自動生成し、ステップＳ１１で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処置がステップＳ１１に進められる。

【０１１７】

［ステップＳ２５］ステップＳ１１で読み込んだメソッド情報と、パラメタ（パラメタ名とパラメタ型）が一致し、メソッド名が不一致の更新前のメソッド情報が、更新前の解析結果情報１３１内にあるか否かを判定する（第５の判定）。該当するメソッド情報があれば、処理がステップＳ２６に進められる。該当するメソッド情報がなければ、処理がステップＳ３２に進められる。

【０１１８】

［ステップＳ２６］ステップＳ２５の判断で該当したメソッド情報が１レコードだけか否かを判断する。１レコードだけ該当した場合には、処理がステップＳ２７に進められる。複数のレコードが該当した場合には、処理がステップＳ２８に進められる。

【０１１９】

［ステップＳ２７］ステップＳ２５で該当すると判断されたメソッド情報に対応するテストパターンにおけるメソッドパラメタの情報を、ステップＳ１１で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処理がステップＳ

２９に進められる。

【０１２０】

［ステップＳ２８］ステップＳ２５で該当すると判断された複数のメソッド情報のうち、メソッド名が一致する文字数が最も多いメソッド情報を引継対象として選択する。そして、選択したメソッド情報に対応するテストパターンにおけるメソッドパラメタの情報を、ステップＳ１１で読み込んだメソッド情報のテストパターンに引き継ぐ。

【０１２１】

［ステップＳ２９］テストパラメタの引継元となったメソッド情報と、ステップＳ１１で読み込んだメソッド情報との戻り値型が一致するか否かを判断する。戻り値型が一致する場合には、処理がステップＳ３０に進められる。戻り値型が一致しない場合には、処理がステップＳ３１に進められる。

【０１２２】

［ステップＳ３０］テストパラメタの引継元となったメソッド情報に対応するテストパターンにおける戻り値の情報を、ステップＳ１１で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処置がステップＳ１１に進められる。

【０１２３】

［ステップＳ３１］メソッド戻り値の情報を自動生成し、ステップＳ１１で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処置がステップＳ１１に進められる。

【０１２４】

［ステップＳ３２］ステップＳ１１で読み込んだメソッド情報へテストパターンを引き継がせる対象となる更新前のメソッド情報が無いと判定し、処理をステップＳ１１に進める。

【０１２５】

図２０は、不一致パラメタ引継処理を示すフローチャートである。以下、図２０に示す処理をステップ番号に沿って説明する。なお、以下の処理は、全て更新時自動引継ぎ部１７０が行う処理である。

【０１２６】

［ステップＳ４１］ステップＳ１１で読み込んだメソッド情報の未処理のパラメタを１つ読み込む。

［ステップＳ４２］処理対象のパラメタが無くなった（パラメタ終了）か否かを判断する。具体的には、ステップＳ４１における読み込み処理で、読み込み対象となる未処理のパラメタが発見されなかったときに、パラメタ終了と判断される。パラメタ終了の場合には、不一致パラメタ引継処理を終了する。パラメタ終了でない（ステップＳ４１でパラメタが読み込まれた）場合には、処理がステップＳ４３に進められる。

【０１２７】

［ステップＳ４３］ステップＳ４１で読み込んだパラメタと、パラメタ名およびパラメタ型が一致するパラメタが、引継対象となっている更新前のメソッド情報に含まれるか否かを判断する（第６の判定）。該当するパラメタがある場合には、処理がステップＳ４４に進められる。該当するパラメタがない場合には、処理がステップＳ４５に進められる。

【０１２８】

［ステップＳ４４］引継対象のメソッド情報に対するテストパターン内のステップＳ４３で該当すると判定されたパラメタの情報を、ステップＳ１１で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処理がステップＳ４１に進められる。

【０１２９】

［ステップＳ４５］ステップＳ４１で読み込んだパラメタと、パラメタ名が一致するパラメタが、引継対象となっている更新前のメソッド情報に含まれるか否かを判断する（第７の判定）。該当するパラメタがある場合には、処理がステップＳ４６に進められる。該当するパラメタがない場合には、処理がステップＳ４７に進められる。

【０１３０】

［ステップＳ４６］パラメタ名が一致し、パラメタ型が不一致の場合の引継処理を実行する。この処理の詳細は後述する。

［ステップＳ４７］ステップＳ４１で読み込んだパラメタと、パラメタ型が一致するパラメタが、引継対象となっている更新前のメソッド情報に含まれるか否かを判断する（第８の判定）。該当するパラメタがある場合には、処理がステップＳ４８に進められる。該当するパラメタがない場合には、処理がステップＳ４９に進められる。

【０１３１】

［ステップＳ４８］引継対象のメソッド情報に対するテストパターン内のステップＳ４７で該当すると判定されたパラメタの情報を、ステップＳ１１で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処理がステップＳ４１に進められる。

【０１３２】

［ステップＳ４９］ステップＳ１１で読み込んだメソッド情報のテストパターンのパラメタに、デフォルト値を設定する。その後、処理がステップＳ４１に進められる。

【０１３３】

図２１は、パラメタ名が一致しパラメタ型が不一致の場合の処理を示すフローチャートである。以下、図２１に示す処理をステップ番号に沿って説明する。なお、以下の処理は、全て更新時自動引継ぎ部１７０が行う処理である。

【０１３４】

［ステップＳ６１］パラメタ型を比較する。更新前のパラメタ型が数値であり更新後のパラメタ型が数値の場合、処理がステップＳ６２に進められる。更新前のパラメタ型が数値であり更新後のパラメタ型が文字型の場合、処理がステップＳ６３に進められる。更新前のパラメタ型がある種の文字型であり更新後のパラメタ型が別種の文字型の場合、処理がステップＳ６４に進められる。その他の場合、処理がステップＳ６５に進められる。

【０１３５】

［ステップＳ６２］数値から数値への変更の場合、更新前のテストパターンのパラメタを、そのまま更新後のテストパターンに引き継がせる。その後、処理が図２０のステップＳ４１に進められる。

【０１３６】

［ステップＳ６３］数値から文字型への変更の場合、更新前のテストパターンのパラメタのパラメタ型を文字型に変更して、更新後のテストパターンに引き継がせる。その後、処理が図２０のステップＳ４１に進められる。

【０１３７】

［ステップＳ６４］ある種の文字型から別種の文字型への変更の場合、更新前のテストパターンのパラメタのパラメタ型を別種の文字型に変更して、更新後のテストパターンに引き継がせる。その後、処理が図２０のステップＳ４１に進められる。

【０１３８】

［ステップＳ６５］その他の場合には、更新後のテストパターンのパラメタとしてデフォルト値を新規に生成する。その後、処理が図２０のステップＳ４１に進められる。

【０１３９】

このようにして、更新前のテストクラス１１１をテストするために作成されたテストパターン１５１の一部または全部が、更新後のテストクラス１１２をテストするためのテストパターン１５２に引き継がれる。

【０１４０】

なお、図１９に示したフローチャートには、５つの判定処理（ステップＳ１３，Ｓ１６，Ｓ１９，Ｓ２２，Ｓ２５）が含まれている。各判定処理では、更新前と更新後とのそれぞれのメソッドの属性の相異点を判定している。

【０１４１】

図２２は、判定処理とメソッドの属性との関係を示す図である。図２２には、判定処理毎に、判定結果がＹＥＳとなるためのメソッドの属性の一致／不一致の関係を示している。属性の一致を丸印（○）で示し、不一致をばつ印（×）で示している。

【０１４２】

第１の判定（ステップＳ１３）では、メソッド名、パラメタ、戻り値が一致した場合に、判定結果がＹＥＳとなる。第２の判定（ステップＳ１６）では、メソ

ッド名、パラメタが一致し、戻り値が不一致の場合に、判定結果がＹＥＳとなる。第３の判定（ステップＳ１９）では、メソッド名、戻り値が一致し、パラメタが不一致の場合に、判定結果がＹＥＳとなる。第４の判定（ステップＳ２２）では、メソッド名のみが一致し、パラメタと戻り値とが不一致の場合に、判定結果がＹＥＳとなる。第５の判定（ステップＳ２５）では、パラメタが一致し、メソッド名が不一致の場合に、判定結果がＹＥＳとなる（戻り値の一致不一致は影響しない）。

【０１４３】

ここで、図７に示した更新前のテストクラス１１１が更新され、図１６に示した更新後のテストクラス１１２になったときのテストパターン引継例を詳しく説明する。

【０１４４】

図２３は、メソッドの変更に応じたテストパターンの自動反映結果の一例を示す図である。

更新後の第１のメソッド１１２ａは、１番目の判定（ステップＳ１３）において、更新前の第１のメソッド１１１ａを引継対象のメソッドとしたときに、判定結果がＹＥＳとなる。その結果、図１０に示したテストパターン群１５１ａ内の全てのテストパターン（識別番号「Ｍ１－１」～「Ｍ１－３」）の内容が引き継がれる。

【０１４５】

更新後の第２のメソッド１１２ｂは、２番目の判定（ステップＳ１６）において、更新前の第２のメソッド１１１ｂを引継対象のメソッドとしたときに、判定結果がＹＥＳとなる。その結果、図１１に示したテストパターン群１５１ｂ内の全てのテストパターン（識別番号「Ｍ２－１」～「Ｍ２－７」）の内容が引き継がれる。

【０１４６】

更新後の第３のメソッド１１２ｃは、３番目の判定（ステップＳ１９）において、更新前の第３のメソッド１１１ｃを引継対象のメソッドとしたときに、判定結果がＹＥＳとなる。その結果、図１２に示したテストパターン群１５１ｃ内の

全てのテストパターン（識別番号「Ｍ３－１」～「Ｍ３－１０」）の内容が引き継がれる。

【０１４７】

更新後の第４のメソッド１１２ｄは、４番目の判定（ステップＳ２２）において、更新前の第４のメソッド１１１ｄを引継対象のメソッドとしたときに、判定結果がＹＥＳとなる。その結果、図１３に示したテストパターン群１５１ｄ内の全てのテストパターン（識別番号「Ｍ４－１」～「Ｍ４－６」）の内容が引き継がれる。

【０１４８】

更新後の第５のメソッド１１２ｅは、５番目の判定（ステップＳ２５）において、更新前の第５のメソッド１１１ｅを引継対象のメソッドとしたときに、判定結果がＹＥＳとなる。その結果、図１４に示したテストパターン群１５１ｅ内の全てのテストパターン（識別番号「Ｍ５－１」～「Ｍ５－１０」）の内容が引き継がれる。

【０１４９】

更新後の第６のメソッド１１２ｆは、全ての判定において、引継対象のメソッドが検出されず、判定結果がＮＯとなる。その結果、更新後の第６のメソッド１１２ｆに関しては、テストパターンの自動引き継ぎは行われない。

【０１５０】

なお、テストパターンの内容が引き継がれるときであっても、必ず全ての内容が引き継がれるのではない。すなわち、パラメタと戻り値との何れかのみが引き継がれる場合もある。以下、図２３に示したメソッドの判定結果毎の具体的なテストパターン引継例を説明する。

【０１５１】

・第１のメソッドのテストパターン引継例

更新後のテストクラス１１２の第１のメソッド１１２ａは、引継対象として更新前のテストクラス１１１の第１のメソッド１１１ａが選択される。更新前の第１のメソッド１１１ａに対するテストパターン群１５１ａは図１０に示す通りである。

【０１５２】

ここで、変更前の第１のメソッド１１１ａと変更後の第１のメソッド１１２ａとを比較する。

図２４は、第１のメソッドの修正内容を示す図である。図２４に示すように、修飾子が「public」から「private」に変更されている。

【０１５３】

図２５は、第１のメソッドの属性毎の比較結果を示す図である。図２５に示すように、その他（修飾子）のみが変更されており、メソッド名、メソッドパラメタ、メソッド戻り値型については、変更されていない。

【０１５４】

この場合、更新前の第１のメソッド１１１ａに対して設定されていたテストパターン群１５１ａ（図１０に示す）がそのまま、更新後の第１のメソッド１１２ａのテストパターン１５２として引き継がれる。

【０１５５】

・第２のメソッドのテストパターン引継例

更新後のテストクラス１１２の第２のメソッド１１２ｂは、引継対象として更新前のテストクラス１１１の第２のメソッド１１１ｂが選択される。更新前の第２のメソッド１１１ｂに対するテストパターン群１５１ｂは図１１に示す通りである。

【０１５６】

ここで、変更前の第２のメソッド１１１ｂと変更後の第２のメソッド１１２ｂとを比較する。

図２６は、第２のメソッドの修正内容を示す図である。図２６に示すように、メソッド戻り値が「EmployeeData」から「EmployeeData[]」に変更されている。

【０１５７】

図２７は、第２のメソッドの属性毎の比較結果を示す図である。図２７に示すように、メソッド戻り値のみが変更されており、メソッド名、メソッドパラメタ、その他の属性については変更されていない。

【０１５８】

この場合、テストパターン群１５１ｂ中の各テストパターンのうち、メソッドパラメタの情報が引き継がれ、メソッド戻り値にはデフォルト値が設定される。

図２８は、更新後の第２のメソッドのテストパターンの例を示す図である。更新後の第２のメソッドのテストパターン群１５２ｂ内の各テストパターンと図１１に示した更新前のテストパターンとの違いは、戻り値がデフォルトの値になったことである。他の情報は、図１１に示した更新前のテストパターンと同じである。図２８の例では、メソッド戻り値のデフォルトの値は、「Null」である。

【０１５９】

・第３のメソッドのテストパターン引継例

更新後のテストクラス１１２の第３のメソッド１１２ｃは、引継対象として更新前のテストクラス１１１の第３のメソッド１１１ｃが選択される。更新前の第３のメソッド１１１ｃに対するテストパターン群１５１ｃは図１２に示す通りである。

【０１６０】

ここで、更新前の第３のメソッド１１１ｃと更新後の第のメソッド１１２ｃとを比較する。

図２９は、第３のメソッドの修正内容を示す図である。図２９に示すように、メソッドパラメタが「String EmployeeName,int Age,int code」から「String EmployeeName,long Age,int section,boolean retire」に変更されている。

【０１６１】

図３０は、第３のメソッドの属性毎の比較結果を示す図である。図３０に示すように、メソッドパラメタのみが変更されており、メソッド名、メソッドパラメタ、その他の属性については変更されていない。

【０１６２】

以下に、パラメタが不一致の場合の判定例を示す。

（第６の判定）　パラメタ名とパラメタ型とが一致するものを引き継ぐ。図３０に示した例によれば、「String EmployeeName」は、更新前後において変更がなく、パラメタ名とパラメタ型とが一致する。したがって、更新前のテストパターンにおける「String EmployeeName」に対応する情報は、そのままを更新後の

テストパターンに引き継がれる。

【０１６３】

（第７の判定）　第６の判定の処理で残存するパラメタに対し、パラメタの名前が一致し、データが引き継ぎ可能なものを引き継ぐ。ここで、データ引継ぎ可能なものとは、パラメタ型が数値→数値の場合、数値→文字列の場合、ある種の文字列→別種の文字列の場合である。数値→数値の場合には、型変換可能な範囲で引き継ぐ（数字系の基本データ型int long double floate decimal BicDecimal）。数字→文字列の場合には、数字を文字列に変換して引き継ぐ。文字列→文字列の場合には、そのまま引き継ぐ。

【０１６４】

図３０に示した例では、更新後の「long age」のパラメタに対して、パラメタ名が一致し、パラメタ型が不一致の更新前の「int age」が検出される。この場合、同じ数字系の場合なので、数字として引き継ぐことができる。そこで、更新前のテストパターンにおいてパラメタ「int age」として設定されていた値が、更新後のテストパターンではパラメタ「long age」の値として引き継がれる。この場合、同じ数字系の場合なので、数字として引き継ぐ。

【０１６５】

（第８の判定）　第７の判定処理で残存するパラメタに対し、パラメタの型が一致するものを引き継ぐ。図３０に示した例では、更新後の「int section」のパラメタに対して、パラメタの型が一致する更新前の「int code」が検出される。そこで、更新前のテストパターンにおいてパラメタ「int code」として設定されていた値が、更新後のテストパターンではパラメタ「int section」の値として引き継がれる。

【０１６６】

第８の判定処理で引継ぎ可能な既存パラメタがみつからない場合は、新規パラメタのテストデータとして、デフォルト値の新規データを生成する。ここで、デフォルト値は、テストデータの入力初期値である。基本型数字の場合、自由な数値が与えられる。図３０に示した例では、「boolean retire」は引き継ぎ対象がないためデフォルト値「true」が生成され、この値を用いてテストパターンが作

成される。

【０１６７】

図３１は、更新後の第３のメソッドのテストパターンの例を示す図である。図３１に示す更新後の第３のメソッドのテストパターン群１５２ｃでは、テストパターンのメソッド名は省略している。また、パラメタの比較判定結果を分かり易くするために、更新前のパラメタ、更新後のパラメタ、および比較結果を表示している。

【０１６８】

更新前のテストパターンにおけるパラメタ「String EmployeeName」の値は、そのまま更新後のテストパターンにおけるパラメタ「String EmployeeName」に引き継がれている。更新前のテストパターンにおけるパラメタ「int Age」の値は、そのまま更新後のテストパターンにおけるパラメタ「long Age」に引き継がれている。更新前のテストパターンにおけるパラメタ「int Code」の値は、そのまま更新後のテストパターンにおけるパラメタ「int Section」に引き継がれている。更新後のテストパターンにおけるパラメタ「Boolean Retire」は、引き継ぎ対象がないため、全てのテストパターンに対してデフォルト値「true」が設定されている。

【０１６９】

・第４のメソッドのテストパターン引継例

更新後のテストクラス１１２の第４のメソッド１１２ｄは、引継対象として更新前のテストクラス１１１の第４のメソッド１１１ｄが選択される。更新前の第４のメソッド１１１ｄに対するテストパターン群１５１ｄは図１３に示す通りである。

【０１７０】

ここで、変更前の第４のメソッド１１１ｄと変更後の第４のメソッド１１２ｄとを比較する。

図３２は、第４のメソッドの修正内容を示す図である。図３２に示すように、メソッド戻り値が「EmployeeData」から「EmployeeData[]」に変更されていると共に、メソッドパラメタが「String EmployeeName,int code」から「String Emp

loyeeName,int section」に変更されている。

【０１７１】

図３３は、第４のメソッドの属性毎の比較結果を示す図である。図３３に示すように、メソッド戻り値とメソッドパラメタが変更されており、メソッド名、その他の属性については変更されていない。

【０１７２】

パラメタのみが不一致の場合と同様に、パラメタ名とパラメタ型との判定が行われ、パラメタ毎の引き継ぎ対象が決定される。

（第６の判定）　パラメタ名とパラメタ型とが一致するパラメタ同士で引き継ぎが行われる。図３３に示した例では、「String EmployeeName」は更新前後において変更がなく、パラメタ名とパラメタ型とが一致する。したがって、更新前のテストパターンにおける「String EmployeeName」に対応する値は、そのままを更新後のテストパターンに引き継がれる。

【０１７３】

（第７の判定）　第６の判定処理で残存するパラメタのうち、パラメタの名前が一致し、データが引き継ぎ可能なものを引き継ぐ。図３３の例では、該当するパラメタはない。

【０１７４】

（第８の判定）　第８の判定処理で残存するパラメタは、パラメタの型が一致するものを引き継ぐ。図３３の例では、更新前のテストパターンにおける「int code」に対応する値が、更新後のテストパターンにおける「int section」に引き継がれる。

【０１７５】

図３４は、更新後の第４のメソッドのテストパターンの例を示す図である。図３４に示す更新後の第４のメソッドのテストパターン群１５２ｄでは、テストパターンのメソッド名は省略している。また、パラメタの比較判定結果を分かり易くするために、更新前のパラメタ、更新後のパラメタ、および比較結果を表示している。

【０１７６】

更新前のテストパターンにおけるパラメタ「String EmployeeName」の値は、そのまま更新後のテストパターンにおけるパラメタ「String EmployeeName」に引き継がれている。更新前のテストパターンにおけるパラメタ「int Code」の値は、そのまま更新後のテストパターンにおけるパラメタ「int Section」に引き継がれている。更新後の各テストパターンのメソッド戻り値は、デフォルトの値（予め設定された初期値）である。

【０１７７】

・第５のメソッドのテストパターン引継例

更新後のテストクラス１１２の第５のメソッド１１２ｅは、引継対象として更新前のテストクラス１１１の第５のメソッド１１１ｅが選択される。更新前の第５のメソッド１１１ｅに対するテストパターン群１５１ｅは図１４に示す通りである。

【０１７８】

ここで、変更前の第５のメソッド１１１ｅと変更後の第５のメソッド１１２ｅとを比較する。

図３５は、第５のメソッドの修正内容を示す図である。図３５に示すように、メソッド名が「updateEmployee」から「updateEmployeebyNo」に変更されている。

【０１７９】

図３６は、第５のメソッドの属性毎の比較結果を示す図である。図３６に示すように、メソッド名のみが変更されており、メソッドパラメタ、メソッド戻り値、その他の属性については変更されていない。

【０１８０】

この場合、パラメタが一致する更新前のメソッドが検索される。すると、更新前の第３のメソッド１１１ｃと第５のメソッド１１１ｅとが検出される。このように、パラメタの一致がＮレコード（Ｎは２以上の整数）ある場合、名前の文字列一致が多いメソッドが引き継ぎ対象となる。この例では、第５のメソッド１１１ｅが引き継ぎ対象となる。したがって、更新前の第５のメソッド１１１ｅに対して設定されていたテストパターン（図１４に示す）のメソッドパラメタの値が

、更新後の第５のメソッド１１２ｅのテストパターンに引き継がれる。また、更新前の第５のメソッド１１１ｅと更新後の第５のメソッド１１２ｅとは、メソッド戻り値が一致している。そのため、更新前の第５のメソッド１１１ｅのテストパターンのメソッド戻り値の値が、更新後の第５のメソッド１１２ｅのテストパターンに引き継がれる。

【０１８１】

図３７は、更新後の第５のメソッドのテストパターンの例を示す図である。図３７に示す更新後の第５のメソッドのテストパターン群１５２ｅでは、パラメタの比較判定結果を分かり易くするために、更新前のパラメタ、更新後のパラメタ、および比較結果を表示している。図３７に示すように、更新前のテストパターンにおけるパラメタ、戻り値が引き継がれ、メソッド名のみが更新後の第５のメソッド１１２ｅのメソッド名に変更されている。

【０１８２】

以上のようにして、更新前のテストクラス１１１をテストするために用意されたテストパターンを用いて、更新後のテストクラス１１２をテストするためのテストパターンを自動生成することができる。このとき、更新後のテストクラス１１２に含まれるメソッドのうち、テストパターンを自動生成できるのは、引き継ぎ対象となる更新前のメソッドが存在するものである。引き継ぎ対象のメソッドが検出されなかった更新後のメソッドについては、手動引継ぎ部１９０を用いて設定することができる。すなわち、作業者からの操作入力に応答して、手動引継ぎ部１９０が更新前のメソッドに対して設定されていたテストパターンを、更新後のメソッドに引き継がせる。

【０１８３】

本実施の形態では、更新後のメソッドに対して、引継対象となる更新前のメソッドを対応付け、テストパターンを引き継がせるための操作入力画面が、ＧＵＩ(Graphical User Interface)により提供される。

【０１８４】

図３８は、引継対象メソッド指定画面の例を示す図である。図３８に示すように、引継対象メソッド指定画面２００には、更新後のメソッド情報表示部２１０

、更新前のメソッド情報表示部２２０、引継候補のないメソッド一覧表示部２３０、上へボタン２０１、下へボタン２０２，削除ボタン２０３、追加ボタン２０４、ＯＫボタン２０５、キャンセルボタン２０６、およびヘルプボタン２０７が設けられている。

【０１８５】

更新後のメソッド情報表示部２１０は、更新後のメソッド情報のリストを表示するための表示領域である。更新後のメソッド情報表示部２１０内には、更新後のクラス名やメソッド名が表示されている。

【０１８６】

更新前のメソッド情報表示部２２０は、更新前のメソッド情報のうち、更新対象として選択されたメソッドのリストを表示するための表示領域である。更新前のメソッド情報表示部２２０内には、更新前のクラス名、メソッド名、ファイル名（テストパターンが登録されているファイル）などが表示されている。

【０１８７】

更新後のメソッド情報表示部２１０と更新前のメソッド情報表示部２２０とは、横方向（同一行）に並べられたメソッド情報同士が互いに関連づけられている。互いに関連するメソッド情報同士で、テストパターンの引継が行われる。なお、更新後のメソッド情報の引継対象の更新前のメソッドが選択されていない場合、その更新後のメソッドと同じ行の更新前のメソッド情報表示部２２０内の欄が、空欄になっている。また、更新前のメソッド情報表示部２２０に表示されている更新前のメソッド情報の登録位置を変更することで、更新前のメソッド情報と更新後のメソッド情報との関連づけを変更することができる。

【０１８８】

引継候補のないメソッド一覧表示部２３０は、更新前のメソッド情報のうち、更新対象として選択されていないメソッド情報（引継候補のないメソッド情報）のリストを表示するための表示領域である。引継候補のないメソッド情報の１つを選択して、更新前のメソッド情報表示部２２０内の空欄の位置に移動することで、引継候補のないメソッド情報を、引継対象のない更新後のメソッド情報に関連づけることができる。更新前のメソッド情報の引継候補のないメソッド一覧表

示部２３０から更新前のメソッド情報表示部２２０への移動は、ドラッグ＆ドロップと呼ばれるマウス操作で指示することができる。

【０１８９】

上へボタン２０１は、更新前のメソッド情報表示部２２０内で選択されている更新前のメソッド情報の登録位置を、上へ移動させるためのボタンである。上へボタン２０１が押されると、更新前のメソッド情報表示部２２０内で選択されている更新前のメソッド情報の位置が１つ上の段へ移動する。このとき、１つ上の段に登録されていた更新前のメソッド情報は、１つ下の段に移動する。

【０１９０】

下へボタン２０２は、更新前のメソッド情報表示部２２０内で選択されている更新前のメソッド情報の登録位置を、下へ移動させるためのボタンである。下へボタン２０２が押されると、更新前のメソッド情報表示部２２０内で選択されている更新前のメソッド情報の位置が１つ下の段へ移動する。このとき、１つ下の段に登録されていた更新前のメソッド情報は、１つ上の段に移動する。

【０１９１】

削除ボタン２０３は、更新前のメソッド情報表示部２２０内で選択されている更新前のメソッド情報の登録を抹消するためのボタンである。削除ボタン２０３が押されると、更新前のメソッド情報表示部２２０内で選択されている更新前のメソッド情報が、更新前のメソッド情報表示部２２０から削除される。更新前のメソッド情報表示部２２０から削除された更新前のメソッド情報は、引継候補のないメソッド一覧表示部２３０内に設定される。

【０１９２】

追加ボタン２０４は、引継候補のないメソッド一覧表示部２３０内で選択されている更新前のメソッド情報を、更新前のメソッド情報表示部２２０内で選択されている欄に追加するためのボタンである。追加ボタン２０４が押されると、引継候補のないメソッド一覧表示部２３０内で選択されている更新前のメソッド情報が、更新前のメソッド情報表示部２２０内で選択されている欄に追加される。更新前のメソッド情報表示部２２０に追加された更新前のメソッド情報は、引継候補のないメソッド一覧表示部２３０から削除される。

【０１９３】

ＯＫボタン２０５は、引継対象メソッド指定画面２００内の情報を確定するためのボタンである。ＯＫボタン２０５が押されると、手動引継ぎ部１９０により、引継対象メソッド指定画面２００に設定されている内容がテストパターン記憶部１５０に登録される。

【０１９４】

キャンセルボタン２０６は、引継対象メソッド指定画面２００内での情報の変更内容を取り消すためのボタンである。キャンセルボタン２０６が押されると、引継対象メソッド指定画面２００内で行われた操作が取り消され、引継対象メソッド指定画面２００が閉じる。

【０１９５】

ヘルプボタン２０７は、操作方法のヘルプ画面を表示させるためのボタンである。ヘルプボタン２０７が押されると、操作方法を案内するヘルプ画面が表示される。

【０１９６】

このように、自動引継ぎ結果の確認と引き継がれなかった情報の手動引継ぎ支持を目的に、作業者が引き継ぎ対象を選択するためのＧＵＩが提供される。これにより、自動引継の対象となる更新前のメソッドが存在しない場合でも、簡単な操作で引継対象の更新前のメソッドを設定することができる。引継対象の更新前のメソッドが指定されれば、そのメソッドのテストパターンが、更新後のメソッドのテストパターンとして引き継がれる。

【０１９７】

また、本実施の形態では、テストパターンの自動引継結果の詳細を画面上で容易に確認することができる。

図３９は、テストパターン引継結果確認画面の例を示す図である。テストパターン引継結果確認画面３００には、テストパターン内容表示部３１０とパラメタ表示部３２０とが設けられている。

【０１９８】

テストパターン内容表示部３１０は、メソッド毎のテストパターンの内容を表

示するための表示領域である。テストパターン内容表示部３１０には、テストパターン毎に、テストパターン番号、テスト内容、正常／異常等の各種情報が表示されている。

【０１９９】

パラメタ表示部３２０は、各テストパターンのパラメタを表示するための表示領域である。パラメタ表示部３２０には、各パラメタのパラメタ名、属性、値等が表示されている。パラメタ表示部３２０において自動生成された値については、強調表示される。たとえば、自動生成された値は、他の値とは異なる色で表示される。図３９の例では、パラメタ「param1」の値「10」が強調表示されている。

【０２００】

このようにテストパターンの引継処理において自動生成された値が強調表示することで、作業者に対して、自動生成された値の確認を促すことができる。作業者は、強調表示された項目の内容を確認し、値の適否を判断する。より適当な値がある場合には、パラメタ表示部３２０に表示されている値を変更することで、パラメタの値を変更することができる。

【０２０１】

以上説明したように本実施の形態によれば、テスト対象のクラスの機能（メソッド）が変更された場合に、設計済みのテストパターンを変更後のメソッドに対応したテストパターンに引き継ぐことができる。

【０２０２】

しかも、機能（メソッド）のインタフェースの属性（メソッド名、メソッドパラメタ、メソッド戻り値など）に着目し、この属性の一致の優先順位のルールにより更新前メソッドと更新後のメソッドの対応つけを自動で行いテストパターンを自動で引き継いでいる。これにより、より適当なテストパターンを引き次ぐことができる。

【０２０３】

さらに、自動引継ぎの候補がなかった場合に、手動で利用者が引き継ぎ対象を指示できるＧＵＩを備えていることにより、引継対象の割り当ての操作が容易と

なる。

【０２０４】

また、自動引継ぎした部分を利用者にわかりやすく通知するＧＵＩを備えていることにより、自動引継の適否を確認するのが容易となる。

また、不一致部分の自動引継ぎ対象外のメソッドパラメタに関してはデフォルト値（初期値）を自動設定するようにしたため、自動引継をしたテストパターンに関しては、そのままの状態で更新後のプログラムの動作テストに利用することができる。

【０２０５】

この際、デフォルト値が設定されたパラメタを強調表示するようにしたため、デフォルト値が設定されたパラメタを容易に認識することができ、適当な他の値に変更するのが容易となる。

【０２０６】

なお、上記の処理機能は、コンピュータによって実現することができる。その場合、テスト支援装置が有すべき機能の処理内容を記述したテスト支援プログラムが提供される。そのプログラムをコンピュータで実行することにより、上記処理機能がコンピュータ上で実現される。処理内容を記述したプログラムは、コンピュータで読み取り可能な記録媒体に記録しておくことができる。コンピュータで読み取り可能な記録媒体としては、磁気記録装置、光ディスク、光磁気記録媒体、半導体メモリなどがある。磁気記録装置には、ハードディスク装置（ＨＤＤ）、フレキシブルディスク（ＦＤ）、磁気テープなどがある。光ディスクには、ＤＶＤ(Digital Versatile Disc)、ＤＶＤ－ＲＡＭ(Random Access Memory)、ＣＤ－ＲＯＭ(Compact Disc Read Only Memory)、ＣＤ－Ｒ(Recordable)／ＲＷ(Re Writable)などがある。光磁気記録媒体には、ＭＯ(Magneto-Optical disc)などがある。

【０２０７】

プログラムを流通させる場合には、たとえば、そのプログラムが記録されたＤＶＤ、ＣＤ－ＲＯＭなどの可搬型記録媒体が販売される。また、プログラムをサーバコンピュータの記憶装置に格納しておき、ネットワークを介して、サーバコ

ンピュータから他のコンピュータにそのプログラムを転送することもできる。

【０２０８】

プログラムを実行するコンピュータは、たとえば、可搬型記録媒体に記録されたプログラムもしくはサーバコンピュータから転送されたプログラムを、自己の記憶装置に格納する。そして、コンピュータは、自己の記憶装置からプログラムを読み取り、プログラムに従った処理を実行する。なお、コンピュータは、可搬型記録媒体から直接プログラムを読み取り、そのプログラムに従った処理を実行することもできる。また、コンピュータは、サーバコンピュータからプログラムが転送される毎に、逐次、受け取ったプログラムに従った処理を実行することもできる。

【０２０９】

（付記１）　開発対象プログラムで定義された機能に対して入力するパラメタと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するためのテスト支援プログラムにおいて、

コンピュータに、

前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述を取得し、

更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述から、前記更新後動作記述と共通性の高い前記更新前動作記述を選択し、

更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターンから、選択された前記更新前動作記述の動作テストのための前記更新前テストパターンを抽出し、

抽出した前記更新前テストパターンの少なくとも一部を引き継いで、前記更新後動作記述の動作テストのための更新後テストパターンを生成する、

処理を実行させることを特徴とするテスト支援プログラム。

【０２１０】

（付記２）　予め指定された複数の属性の情報の少なくとも１つが前記更新後動作記述と同じ値の前記更新前動作記述を、共通性の高い前記更新前動作記述と

して選択することを特徴とする付記１記載のテスト支援プログラム。

【０２１１】

（付記３）　前記複数の属性に優先順が設定されており、高い優先順の属性の情報が同一である前記更新前動作記述を優先的に選択することを特徴とする付記２記載のテスト支援プログラム。

【０２１２】

（付記４）　前記複数の属性として、機能を特定する機能名、機能に入力するパラメタの種別、機能が処理を実行後の出力する戻り値の種別を含むことを特徴とする付記２記載のテスト支援プログラム。

【０２１３】

（付記５）　前記更新後動作記述と選択された前記更新前動作記述とで共通した情報の属性に応じて、抽出した前記更新前テストパターンから引き継ぐべき情報を決定することを特徴とする付記１記載のテスト支援プログラム。

【０２１４】

（付記６）　前記複数の属性として、機能を特定する機能名、機能に入力するパラメタの種別、および機能が処理を実行後の出力する戻り値の種別が指定されており、機能名が一致した場合、他の同一属性に対応するデータを、前記更新前テストパターンから引き継ぐことを特徴とする付記５記載のテスト支援プログラム。

【０２１５】

（付記７）　前記更新後テストパターンのうち、前記更新前テストパターンから引き継がない情報を自動生成することを特徴とする付記１記載のテスト支援プログラム。

【０２１６】

（付記８）　前記自動生成した情報を強調して、前記更新後テストパターンの内容を表示することを特徴とする付記７記載のテスト支援プログラム。

（付記９）　共通性の高い前記更新前動作記述がない場合、操作入力により指定された前記更新前動作記述を選択することを特徴とする付記１記載のテスト支援プログラム。

【０２１７】

（付記１０）　選択されなかった前記更新前動作記述の一覧を表示し、操作入力により前記一覧内で指定された前記更新前動作記述を選択することを特徴とする付記９記載のテスト支援プログラム。

【０２１８】

（付記１１）　オブジェクト指向で作成された開発対象プログラムで定義された機能に対して入力するパラメタと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するためのテスト支援プログラムにおいて、

コンピュータに、

前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後メソッド情報を取得し、

更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前メソッド情報から、前記更新後メソッド情報と共通性の高い前記更新前メソッド情報を選択し、

更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターンから、選択された前記更新前メソッド情報の動作テストのための前記更新前テストパターンを抽出し、

抽出した前記更新前テストパターンの少なくとも一部を引き継いで、前記更新後メソッド情報の動作テストのための更新後テストパターンを生成する、

処理を実行させることを特徴とするテスト支援プログラム。

【０２１９】

（付記１２）　開発対象プログラムで定義された機能に対して入力するパラメタと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するためのテスト支援方法において、

前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述を取得し、

更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述から、前記更新後動作記述と共通性の高い前記更新前動作記述を

選択し、

更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターンから、選択された前記更新前動作記述の動作テストのための前記更新前テストパターンを抽出し、

抽出した前記更新前テストパターンの少なくとも一部を引き継いで、前記更新後動作記述の動作テストのための更新後テストパターンを生成する、

ことを特徴とするテスト支援方法。

【０２２０】

（付記１３）　開発対象プログラムで定義された機能に対して入力するパラメタと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するテスト支援装置において、

前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述を取得する更新後動作記述取得手段と、

更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述を記憶する更新前動作記述記憶手段と、

前記更新前動作記述記憶手段から、更新後動作記述取得手段で取得された前記更新後動作記述と共通性の高い前記更新前動作記述を選択する更新前動作記述選択手段と、

更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターンを記憶する更新前テストパターン記憶手段と、

更新前テストパターン記憶手段から、前記更新前動作記述選択手段で選択された前記更新前動作記述の動作テストのための前記更新前テストパターンを抽出する更新前テストパターン抽出手段と、

前記更新前テストパターン抽出手段で抽出した前記更新前テストパターンの少なくとも一部を引き継いで、前記更新後動作記述の動作テストのための更新後テストパターンを生成する更新後テストパターン生成手段と、

を有することを特徴とするテスト支援装置。

【０２２１】

（付記１４）　開発対象プログラムで定義された機能に対して入力するパラメタと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するためのテスト支援プログラムを記録したコンピュータ読み取り可能な記録媒体において、

前記コンピュータに、

前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述を取得し、

更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述から、前記更新後動作記述と共通性の高い前記更新前動作記述を選択し、

更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターンから、選択された前記更新前動作記述の動作テストのための前記更新前テストパターンを抽出し、

抽出した前記更新前テストパターンの少なくとも一部を引き継いで、前記更新後動作記述の動作テストのための更新後テストパターンを生成する、

処理を実行させることを特徴とするテスト支援プログラムを記録したコンピュータ読み取り可能な記録媒体。

【０２２２】

【発明の効果】

以上説明したように本発明では、更新後動作記述と共通性の高い更新前動作記述に対して設定された更新前テストパターンの内容を引き継いで、更新後動作記述に対する更新後テストパターンを生成するようにしたため、開発対象プログラムが更新されたときにもテストパターンを一から作り直す必要が無くなり、テストパターンの作成が容易となる。

【図面の簡単な説明】

【図１】

実施の形態に適用される発明の概念図である。

【図２】

本発明の実施の形態に用いるコンピュータのハードウェア構成例を示す図であ

る。

【図３】

テスト支援装置の有する処理機能を示すブロック図である。

【図４】

テストパターン自動引き継ぎの入出力関係を示す図である。

【図５】

メソッドに含まれる情報の属性を示す図である。

【図６】

解析結果情報の保存形式例を示す図である。

【図７】

更新前のテストクラス例を示す図である。

【図８】

更新前のメソッド情報テーブルの例を示す図である。

【図９】

更新前のメソッドパラメタテーブルの例を示す図である。

【図１０】

テストクラス変更前の第１のメソッドのテストパターンの例を示す図である。

【図１１】

テストクラス変更前の第２のメソッドのテストパターンの例を示す図である。

【図１２】

テストクラス変更前の第３のメソッドのテストパターンの例を示す図である。

【図１３】

テストクラス変更前の第４のメソッドのテストパターンの例を示す図である。

【図１４】

テストクラス変更前の第５のメソッドのテストパターンの例を示す図である。

【図１５】

テストクラス変更前の第６のメソッドのテストパターンの例を示す図である。

【図１６】

更新後のテストパターンの例を示す図である。

【図１７】

更新後のメソッド情報テーブルの例を示す図である。

【図１８】

更新後のメソッドパラメタテーブルの例を示す図である。

【図１９】

テストパターン自動引継処理の基本的なメインのフローチャートである。

【図２０】

不一致パラメタ引継処理を示すフローチャートである。

【図２１】

パラメタ名が一致しパラメタ型が不一致の場合の処理を示すフローチャートである。

【図２２】

判定処理とメソッドの属性との関係を示す図である。

【図２３】

メソッドの変更に応じたテストパターンの自動反映結果の一例を示す図である。

【図２４】

第１のメソッドの修正内容を示す図である。

【図２５】

第１のメソッドの属性毎の比較結果を示す図である。

【図２６】

第２のメソッドの修正内容を示す図である。

【図２７】

第２のメソッドの属性毎の比較結果を示す図である。

【図２８】

更新後の第２のメソッドのテストパターンの例を示す図である。

【図２９】

第３のメソッドの修正内容を示す図である。

【図３０】

第３のメソッドの属性毎の比較結果を示す図である。

【図３１】

更新後の第３のメソッドのテストパターンの例を示す図である。

【図３２】

第４のメソッドの修正内容を示す図である。

【図３３】

第４のメソッドの属性毎の比較結果を示す図である。

【図３４】

更新後の第４のメソッドのテストパターンの例を示す図である。

【図３５】

第５のメソッドの修正内容を示す図である。

【図３６】

第５のメソッドの属性毎の比較結果を示す図である。

【図３７】

更新後の第５のメソッドのテストパターンの例を示す図である。

【図３８】

引継対象メソッド指定画面の例を示す図である。

【図３９】

テストパターン引継結果確認画面の例を示す図である。

【符号の説明】

１　更新後動作記述

２　更新前動作記述

３　更新前テストパターン

４　更新後テストパターン

１００　テスト支援装置

１１０　テストクラス記憶部

１２０　構文・意味解析部

１３０　第１の解析結果記憶部

１４０　テストパターン作成部

１５０　テストパターン記憶部

１６０　第２の解析結果記憶部

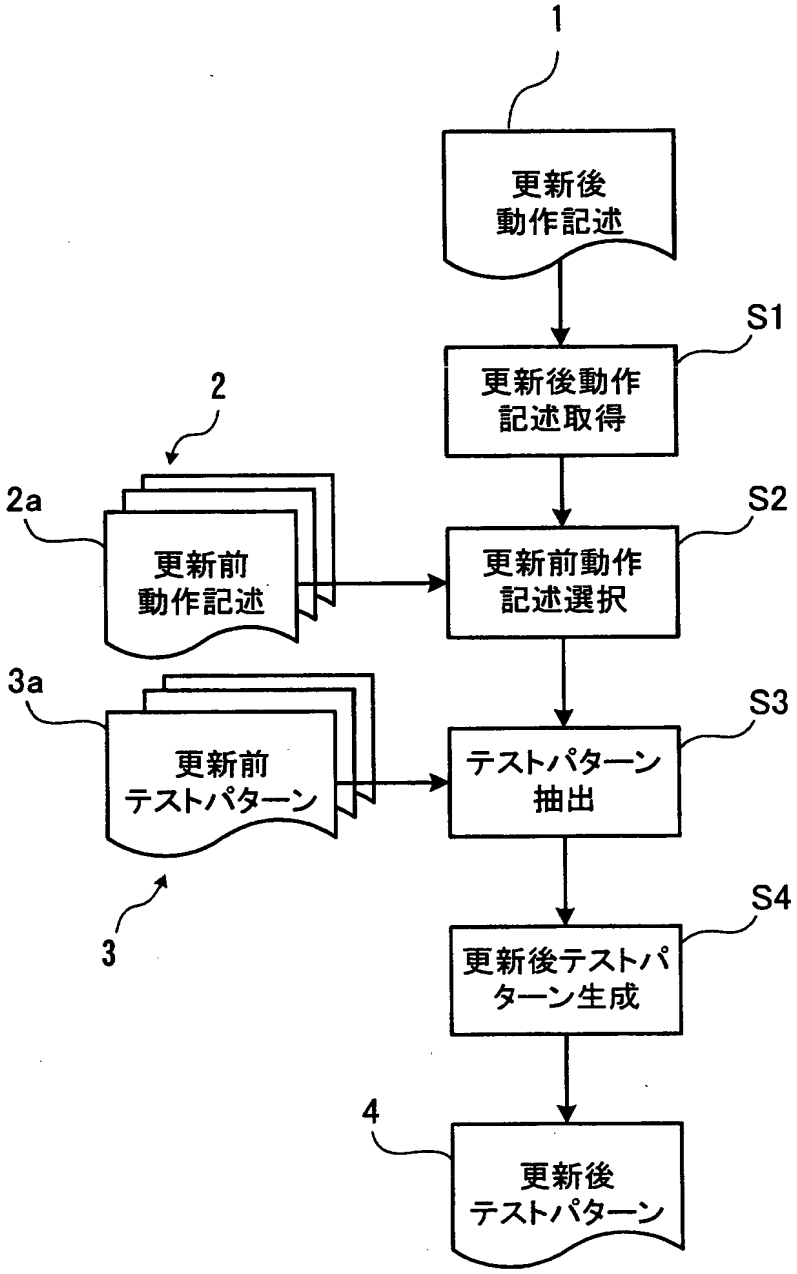
１７０　更新時自動引継ぎ部

１８０　テスト実施部

１９０　手動引継ぎ部

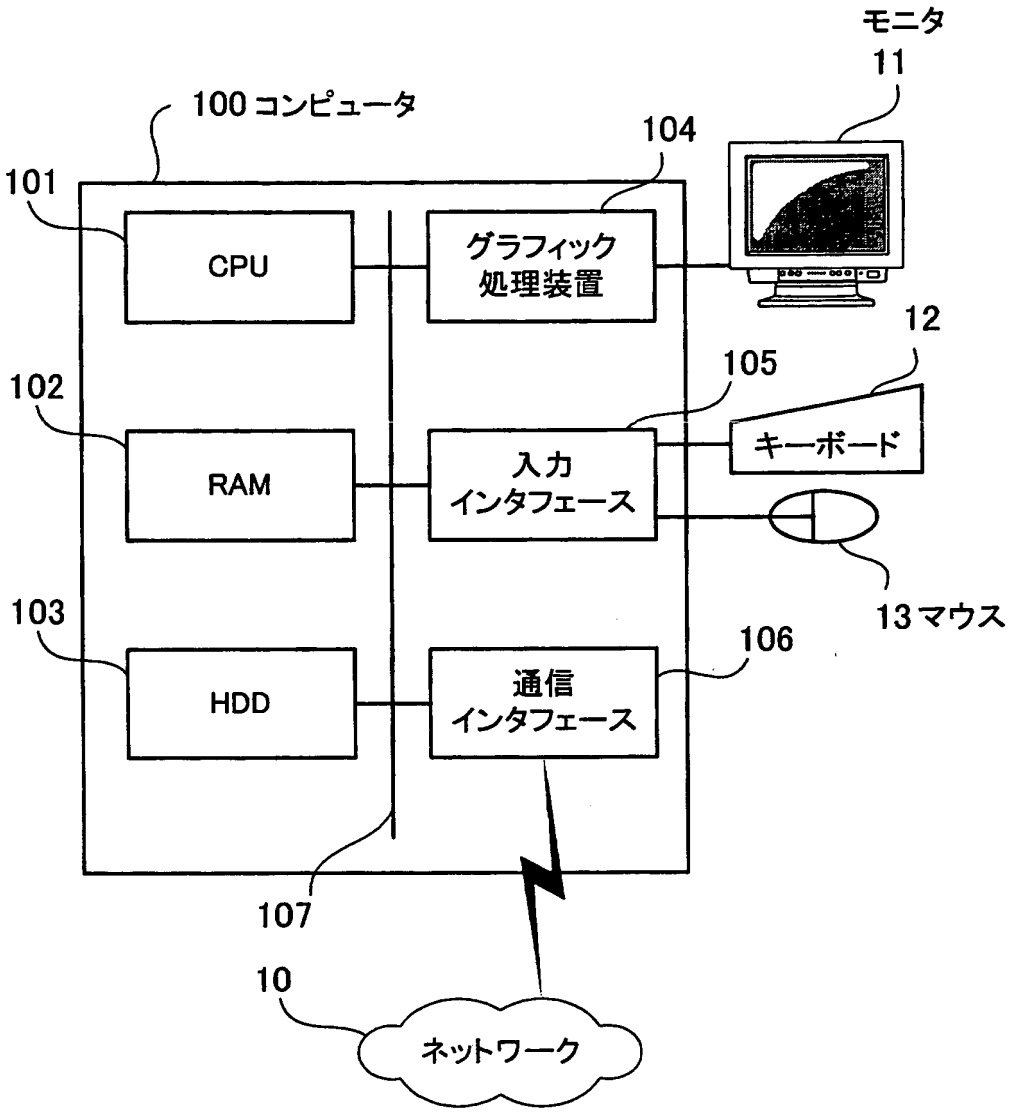
【書類名】　　図面

【図１】



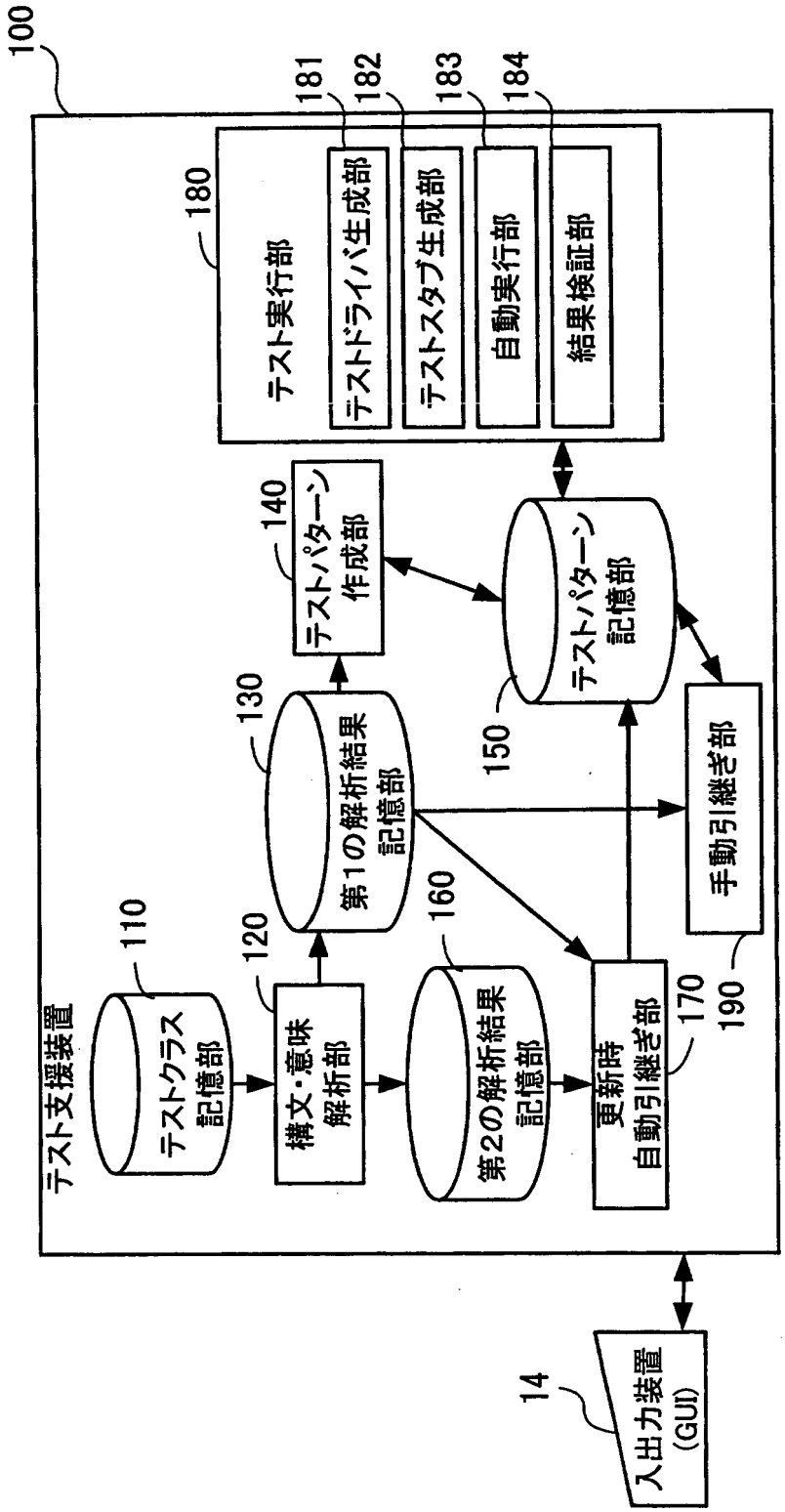
出証特２００３－３０１３４９７

【図２】

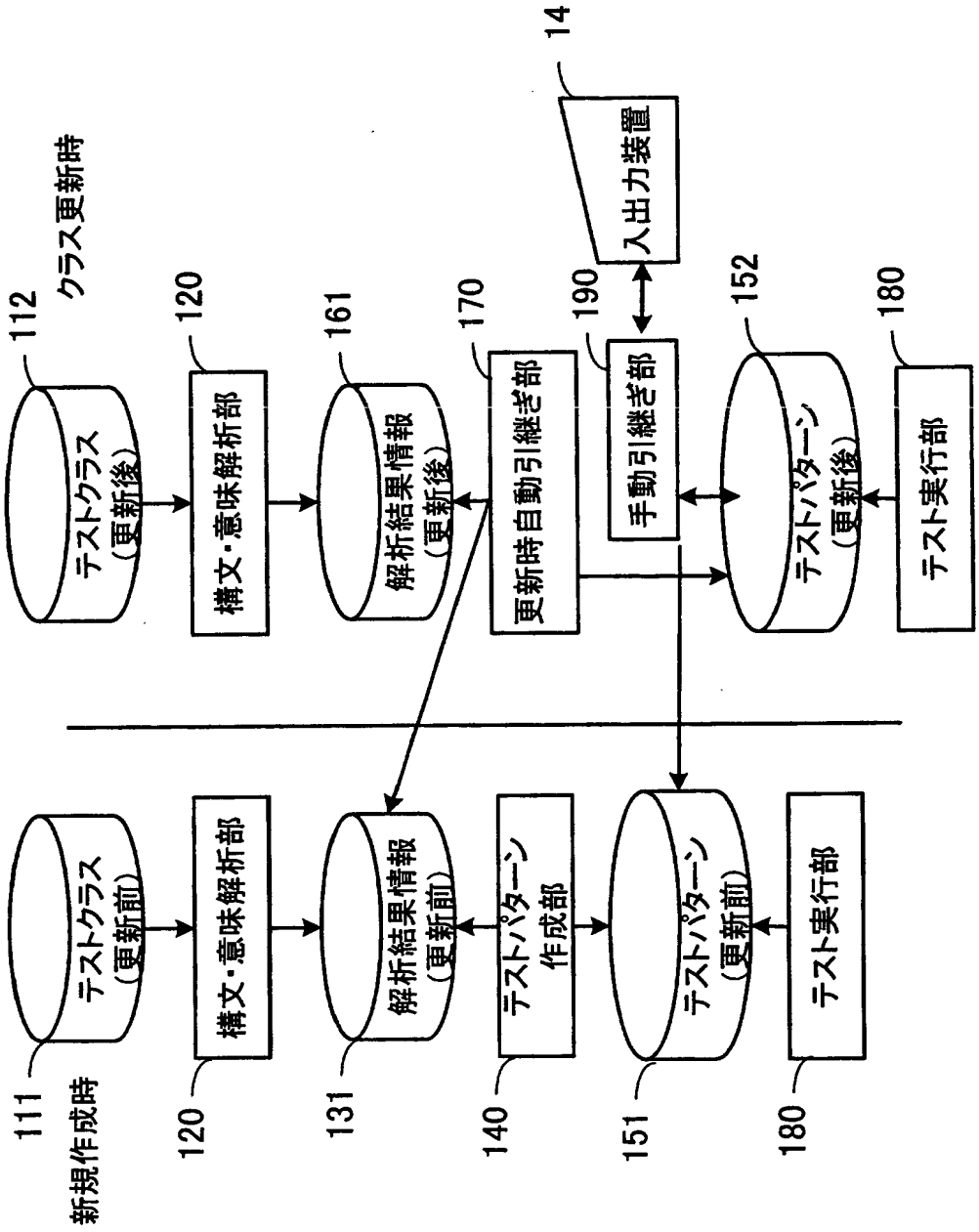


出証特２００３－３０１３４９７

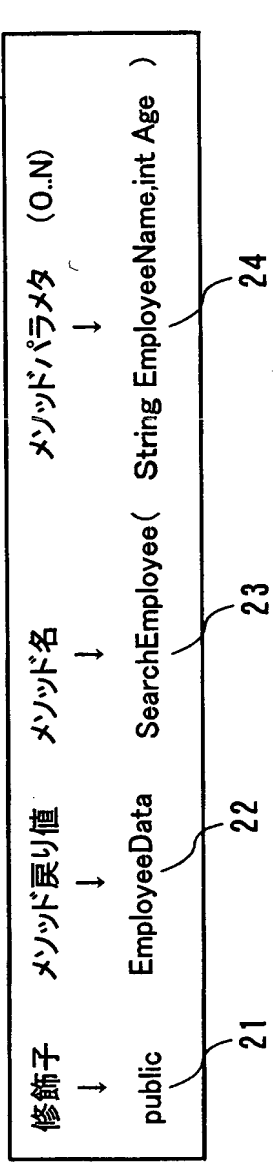
【図３】


100
テスト支援装置
テストクラス記憶部
110
構文・意味解析部
120
第1の解析結果記憶部
130
テストパターン作成部
140
第2の解析結果記憶部
160
更新時自動引継ぎ部
170
テストパターン記憶部
150
手動引継ぎ部
190
テスト実行部
180
テストドライバ生成部
181
テストスタブ生成部
182
自動実行部
183
結果検証部
184
入出力装置(GUI)
14

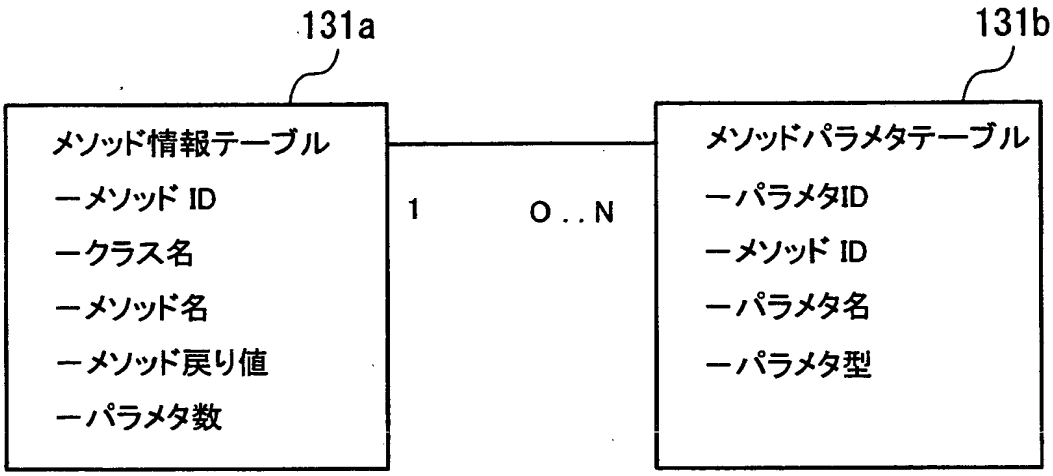
【図４】


クラス更新時
112
テストクラス（更新後）
120
構文・意味解析部
161
解析結果情報（更新後）
170
更新時自動引継ぎ部
14
入出力装置
190
手動引継ぎ部
152
テストパターン（更新後）
180
テスト実行部
新規作成時
111
テストクラス（更新前）
120
構文・意味解析部
131
解析結果情報（更新前）
140
テストパターン作成部
151
テストパターン（更新前）
180
テスト実行部

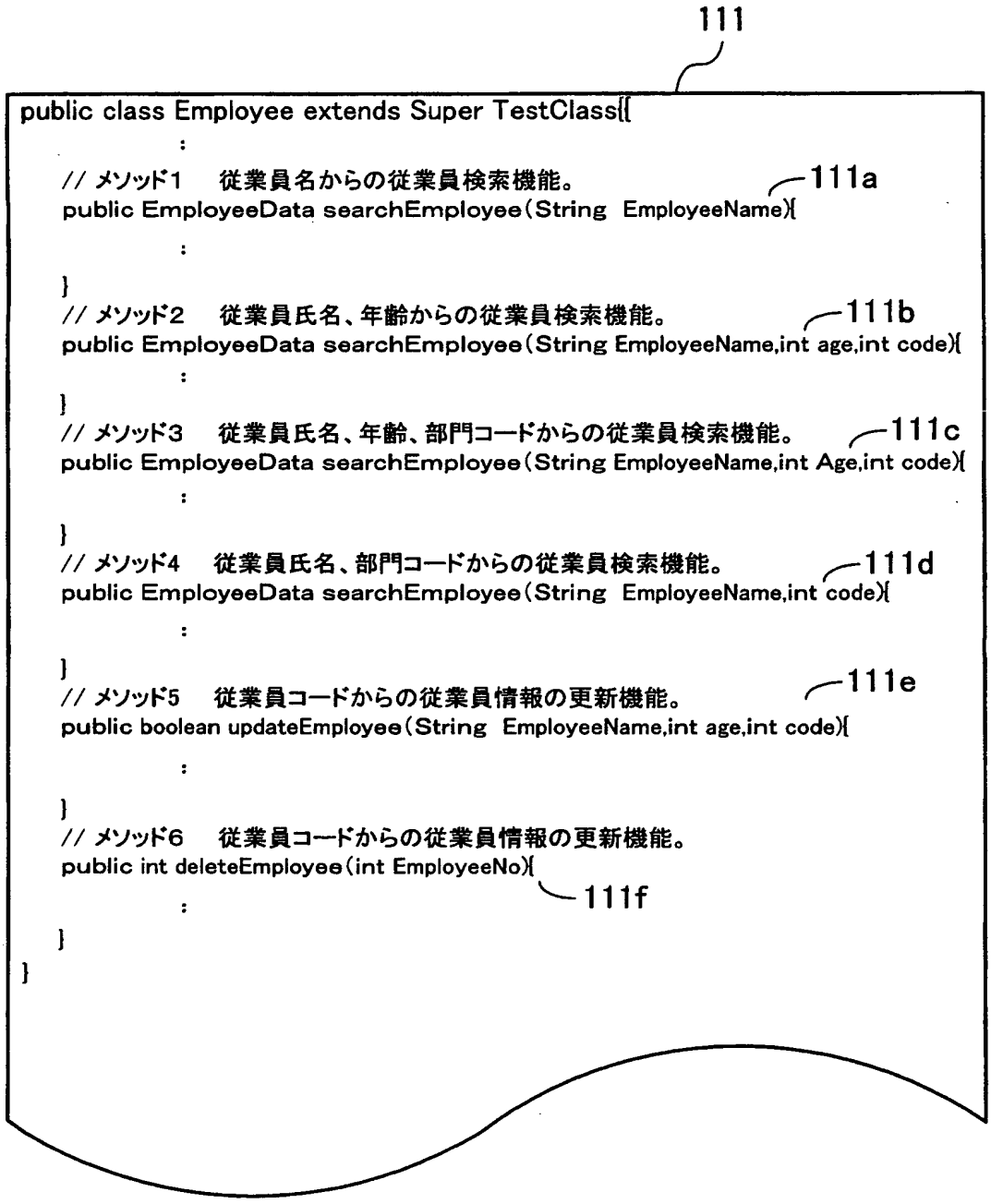
【図５】


20
修飾子
メソッド戻り値
メソッド名
メソッドパラメタ (0..N)
public
EmployeeData
SearchEmployee(String EmployeeName,int Age)
21
22
23
24

【図６】


131a
131b
メソッド情報テーブル
－メソッド ID
－クラス名
－メソッド名
－メソッド戻り値
－パラメタ数
1
O . . N
メソッドパラメタテーブル
－パラメタID
－メソッド ID
－パラメタ名
－パラメタ型

【図７】


111
public class Employee extends Super TestClass{{
:
// メソッド1　従業員名からの従業員検索機能。
111a
public EmployeeData searchEmployee(String EmployeeName){
:
}
// メソッド2　従業員氏名、年齢からの従業員検索機能。
111b
public EmployeeData searchEmployee(String EmployeeName,int age,int code){
:
}
// メソッド3　従業員氏名、年齢、部門コードからの従業員検索機能。
111c
public EmployeeData searchEmployee(String EmployeeName,int Age,int code){
:
}
// メソッド4　従業員氏名、部門コードからの従業員検索機能。
111d
public EmployeeData searchEmployee(String EmployeeName,int code){
:
}
// メソッド5　従業員コードからの従業員情報の更新機能。
111e
public boolean updateEmployee(String EmployeeName,int age,int code){
:
}
// メソッド6　従業員コードからの従業員情報の更新機能。
public int deleteEmployee(int EmployeeNo){
111f
:
}
}

【図８】

131a

メソッドID	クラス名	メソッド名	戻り値型	パラメタ数
1	Employee	searchEmployee	EmployeeData	1
2	Employee	searchEmployee	EmployeeData	2
3	Employee	searchEmployee	EmployeeData	3
4	Employee	searchEmployee	EmployeeData	2
5	Employee	updateEmployee	boolean	3
6	Employee	deleteEmployee	int	1

【図９】

131b

パラメタID	メソッドID	パラメタ名	パラメタ名
1	1	EmployeeName	String
2	2	EmployeeName	String
3	2	age	int
4	3	EmployeeName	String
5	3	age	int
6	3	code	int
7	4	EmployeeName	String
8	4	code	int
9	5	EmployeeName	String
10	5	age	int
11	5	code	int
12	6	EmployeeNo	int

【図１０】

151a

No	メソッド名	パラメタ	正常/異常	戻り値	テストの観点
M1	searchEmployee	EmployeeName		Employee Data	
M1-1	searchEmployee	“F通 太郎”	正常	検索結果	正常系のテスト
M1-2	searchEmployee	“”	異常	Null	氏名が空白の場合の異常系
M1-3	searchEmployee	Null	異常	Null	氏名がNullの場合の異常系

【図１１】

151b

No	メソッド名	パラメタ		正常/異常	戻り値	テストの観点
		EmployeeName	Age		Employee Data	
M2	searchEmployee					
M2-1	searchEmployee	“F通　太郎”	29	正常	検索結果	正常系のテスト
M2-2	searchEmployee	“”	29	異常	Null	氏名が空白の場合の異常系
M2-3	searchEmployee	Null	29	異常	Null	氏名がNullの場合の異常系
M2-4	searchEmployee	“F通　花子”	0	異常	Null	年齢が0の場合の異常系
M2-5	searchEmployee	“F通　花子”	10	異常	Null	年齢が従業員雇用範囲外の場合
M2-6	searchEmployee	“F通　花子”	65	異常	Null	年齢が従業員雇用範囲外の場合
M2-7	searchEmployee	“F通　花子”	Null	異常	Null	年齢がNullの場合

【図１２】

151c

No	メソッド名	パラメタ			正常/異常	戻り値	テストの観点
		EmployeeName	Age	code			
M3	searchEmployee	“F通　太郎”	29	12345	正常	検索結果	正常系のテスト
M3-1	searchEmployee	“”	29	12345	異常	Null	氏名が空白の場合の異常系
M3-2	searchEmployee	Null	29	12345	異常	Null	氏名がNullの場合の異常系
M3-3	searchEmployee	“F通　花子”	0	12345	異常	Null	年齢が0の場合の異常系
M3-4	searchEmployee	“F通　花子”	10	12345	異常	Null	年齢が従業員雇用範囲外の場合
M3-5	searchEmployee	“F通　花子”	65	12345	異常	Null	年齢が従業員雇用範囲外の場合
M3-6	searchEmployee	“F通　花子”	Null	12345	異常	Null	年齢がNullの場合
M3-7	searchEmployee	“F通　花子”	29	0	異常	Null	従業員コードが0の場合
M3-8	searchEmployee	“F通　花子”	29	Null	異常	Null	従業員コードがNullの場合
M3-9	searchEmployee	“F通　花子”	Null	Null	異常	Null	氏名、年齢がNullの場合
M3-10	searchEmployee	Null	Null	Null	異常	Null	

【図１３】

151d

No	メソッド名	パラメタ		正常/異常	戻り値	テストの観点
M4-1	searchEmployee	EmployeeName	code			
M4-1	searchEmployee	“F通　太郎”	12345	正常	検索結果	正常系のテスト
M4-2	searchEmployee	“”	12345	異常	Null	氏名が空白の場合の異常系
M4-3	searchEmployee	Null	12345	異常	Null	氏名がNullの場合の異常系
M4-4	searchEmployee	“F通　花子”	0	異常	Null	部コードが0の場合
M4-5	searchEmployee	“F通　花子”	Null	異常	Null	部コードがNullの場合
M4-6	searchEmployee	Null	Null	異常	Null	氏名、年齢がNullの場合

【図１４】

151e

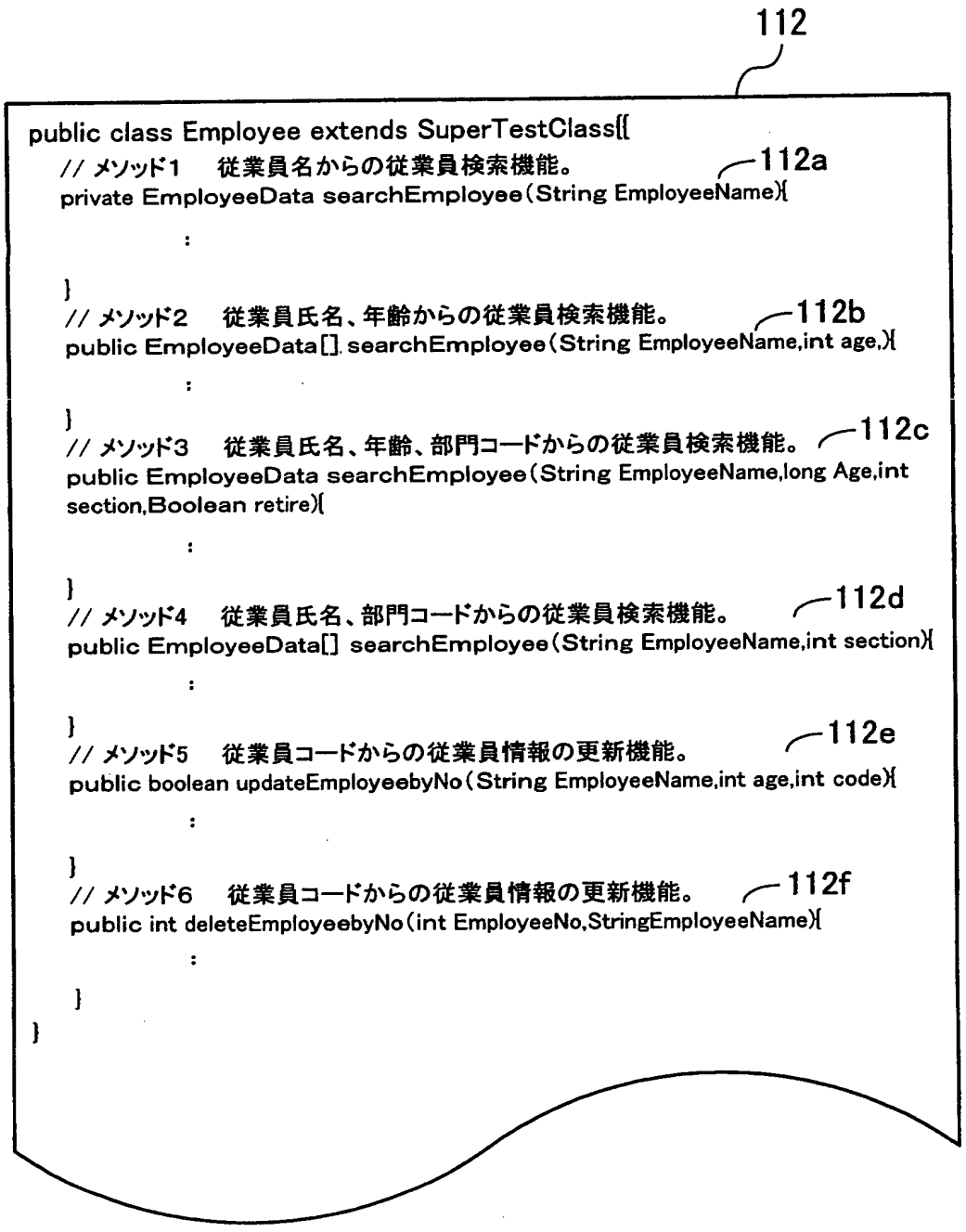
No	メソッド名	パラメタ			正常/異常	戻り値	テストの観点
		EmployeeName	Age	code			
M5	updateEmployee						
M5-1	updateEmployee	“F通 太郎”	29	12345	正常	検索結果	正常系のテスト
M5-2	updateEmployee	“”	29	12345	異常	Null	氏名が空白の場合の異常系
M5-3	updateEmployee	Null	29	12345	異常	Null	氏名がNullの場合の異常系
M5-4	updateEmployee	“F通 花子”	0	12345	異常	Null	年齢が0の場合の異常系
M5-5	updateEmployee	“F通 花子”	10	12345	異常	Null	年齢が従業員雇用範囲外の場合
M5-6	updateEmployee	“F通 花子”	65	12345	異常	Null	年齢が従業員雇用範囲外の場合
M5-7	updateEmployee	“F通 花子”	Null	12345	異常	Null	年齢がNullの場合
M5-8	updateEmployee	“F通 花子”	29	0	異常	Null	従業員コードが0の場合
M5-9	updateEmployee	“F通 花子”	29	Null	異常	Null	従業員コードがNullの場合
M5-10	updateEmployee	Null	Null	Null	異常	Null	従業員コード、氏名、年齢がNullの場合

【図１５】

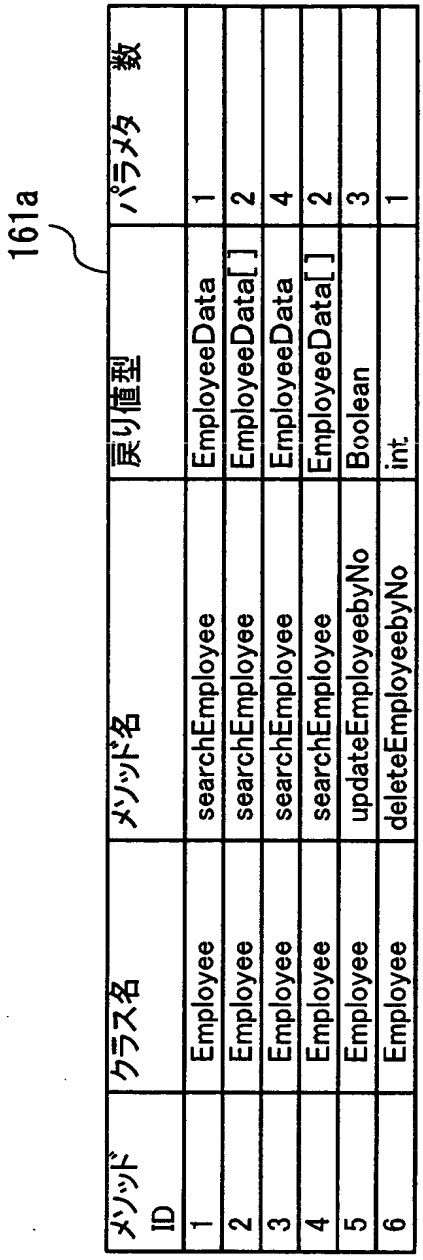
151f

No	メソッド名	パラメタ	正常/異常	戻り値	テストの観点
M6	deleteEmployee	code		Boolean	
M6-1	deleteEmployee	12345	正常	True	正常系のテスト
M6-2	deleteEmployee	0	異常	False	従業員コードが0の場合の異常系
M6-3	deleteEmployee	Null	異常	False	従業員コードがNullの場合の異常系
M6-4	deleteEmployee	11111	異常	False	従業員コードが該当なしの場合の異常系

【図１６】


112
public class Employee extends SuperTestClass{{
// メソッド1　従業員名からの従業員検索機能。
112a
private EmployeeData searchEmployee(String EmployeeName){
:
}
// メソッド2　従業員氏名、年齢からの従業員検索機能。
112b
public EmployeeData[] searchEmployee(String EmployeeName,int age,){
:
}
// メソッド3　従業員氏名、年齢、部門コードからの従業員検索機能。
112c
public EmployeeData searchEmployee(String EmployeeName,long Age,int section,Boolean retire){
:
}
// メソッド4　従業員氏名、部門コードからの従業員検索機能。
112d
public EmployeeData[] searchEmployee(String EmployeeName,int section){
:
}
// メソッド5　従業員コードからの従業員情報の更新機能。
112e
public boolean updateEmployeebyNo(String EmployeeName,int age,int code){
:
}
// メソッド6　従業員コードからの従業員情報の更新機能。
112f
public int deleteEmployeebyNo(int EmployeeNo,StringEmployeeName){
:
}
}

【図１７】



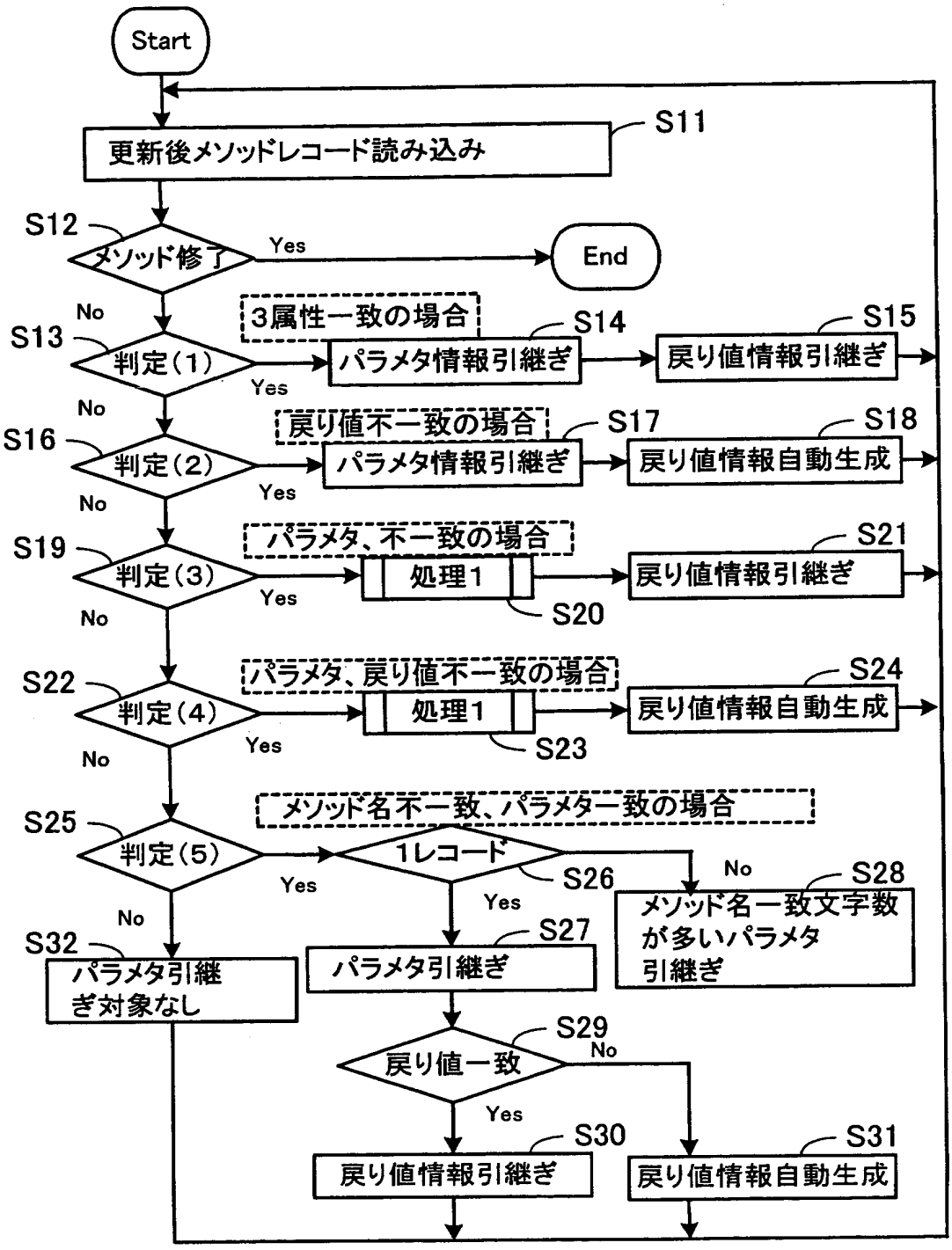
メソッドID	クラス名	メソッド名	戻り値型	パラメタ数
1	Employee	searchEmployee	EmployeeData	1
2	Employee	searchEmployee	EmployeeData[ ]	2
3	Employee	searchEmployee	EmployeeData	4
4	Employee	searchEmployee	EmployeeData[ ]	2
5	Employee	updateEmployeebyNo	Boolean	3
6	Employee	deleteEmployeebyNo	int	1

【図１８】

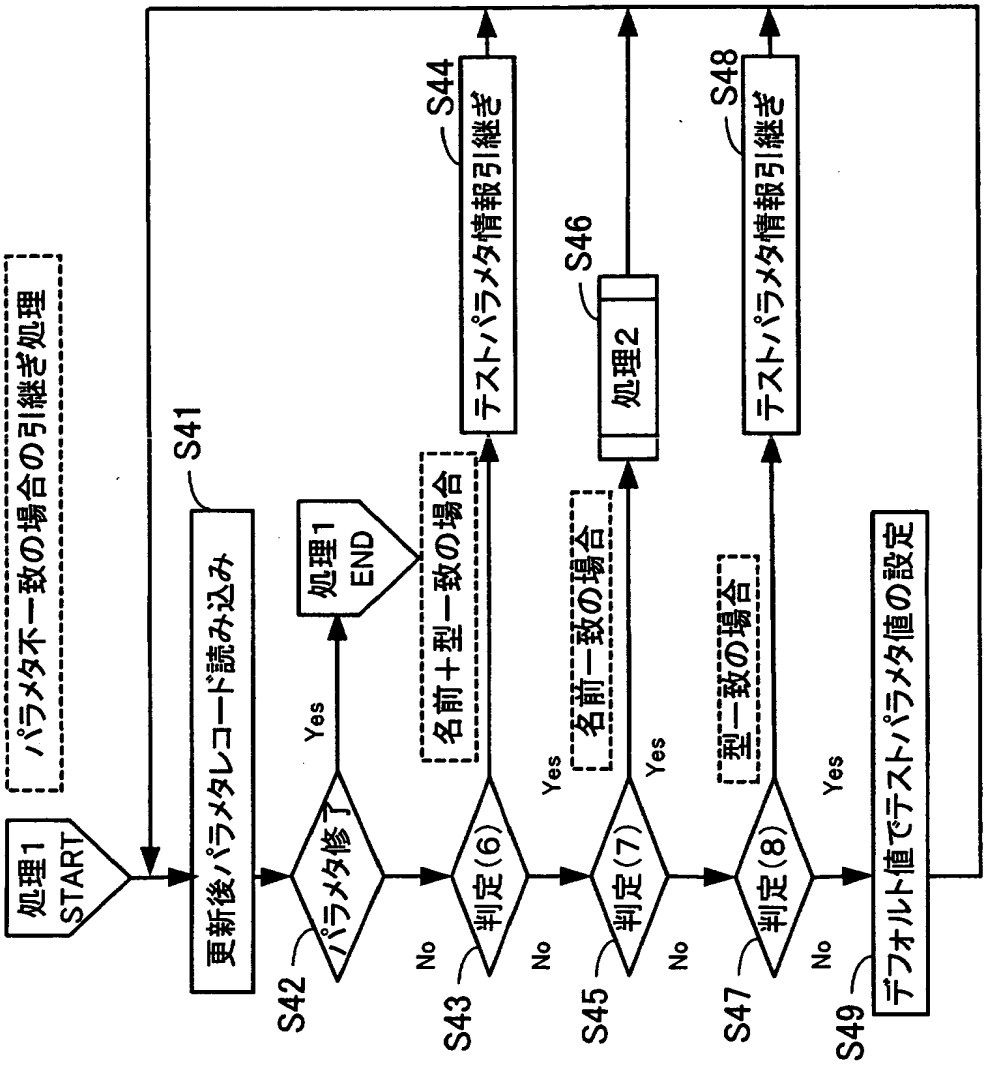
161b

パラメタID	メソッドID	パラメタ名	パラメタ名
1	1	EmployeeName	String
2	2	EmployeeName	String
3	2	Age	int
4	3	EmployeeName	String
5	3	Age	Long
6	3	Section	int
7	3	Retire	Boolean
8	4	EmployeeName	String
9	4	Section	int
10	5	EmployeeName	String
11	5	Age	int
12	5	Code	int
13	6	EmployeeNo	int
14	6	EmployeeName	String

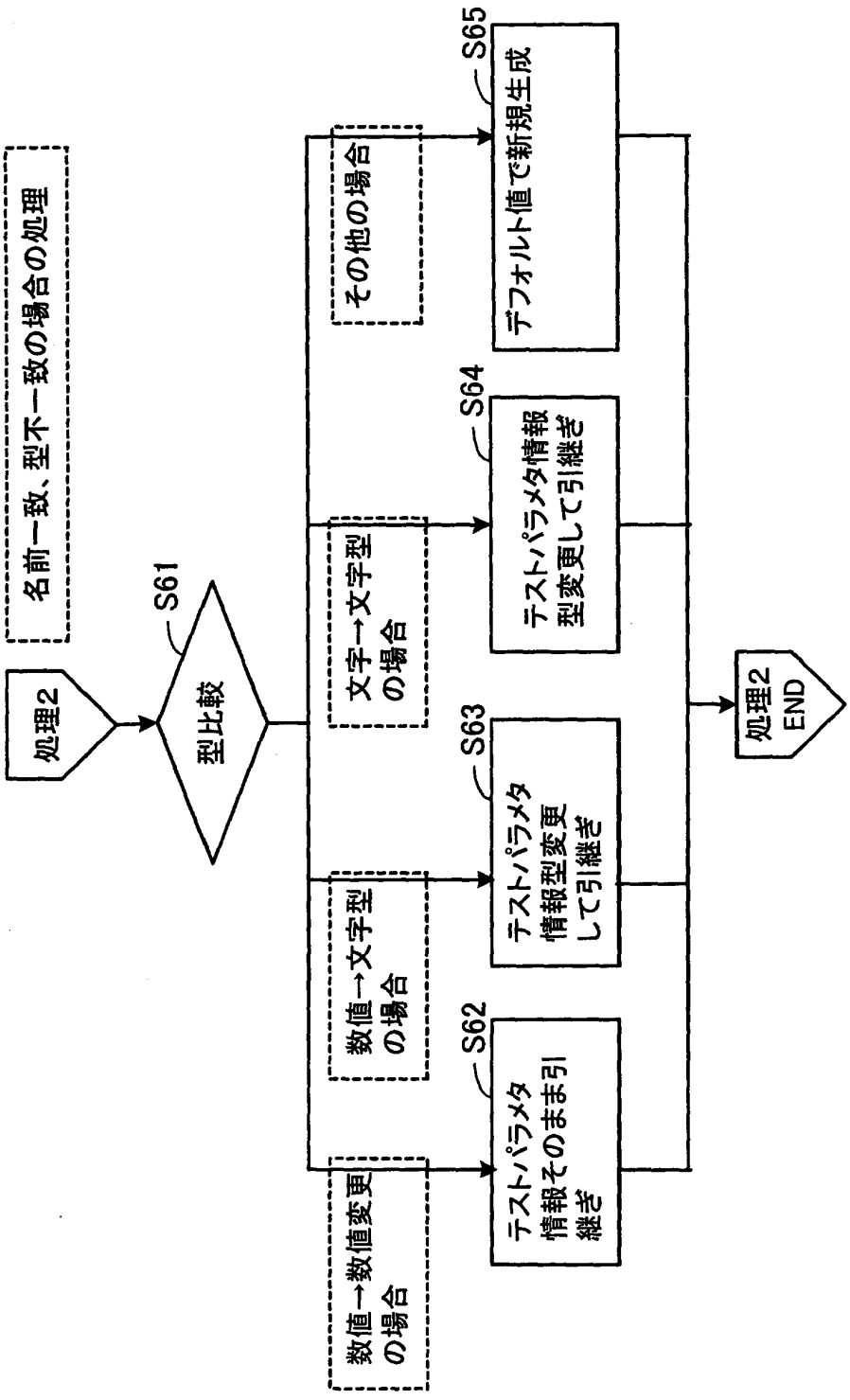
【図１９】


Start
更新後メソッドレコード読み込み
S11
S12
メソッド修了
Yes
End
No
S13
判定(1)
3属性一致の場合
S14
パラメタ情報引継ぎ
S15
戻り値情報引継ぎ
Yes
No
S16
判定(2)
戻り値不一致の場合
S17
パラメタ情報引継ぎ
S18
戻り値情報自動生成
Yes
No
S19
判定(3)
パラメタ、不一致の場合
処理1
S20
S21
戻り値情報引継ぎ
Yes
No
S22
判定(4)
パラメタ、戻り値不一致の場合
処理1
S23
S24
戻り値情報自動生成
Yes
No
S25
判定(5)
メソッド名不一致、パラメター致の場合
1レコード
S26
No
S28
メソッド名一致文字数が多いパラメタ引継ぎ
Yes
Yes
No
S32
パラメタ引継ぎ対象なし
S27
パラメタ引継ぎ
S29
戻り値一致
No
Yes
S30
戻り値情報引継ぎ
S31
戻り値情報自動生成

【図２０】


処理1
START
パラメタ不一致の場合の引継ぎ処理
S41
更新後パラメタレコード読み込み
S42
パラメタ修了
Yes
処理1
END
No
S43
判定(6)
名前＋型一致の場合
S44
テストパラメタ情報引継ぎ
Yes
No
S45
判定(7)
名前一致の場合
S46
処理2
Yes
No
S47
判定(8)
型一致の場合
S48
テストパラメタ情報引継ぎ
Yes
No
S49
デフォルト値でテストパラメタ値の設定

【図２１】


処理2
名前一致、型不一致の場合の処理
S61
型比較
文字→文字型の場合
数値→文字型の場合
数値→数値変更の場合
その他の場合
S62
テストパラメタ情報そのまま引継ぎ
S63
テストパラメタ情報型変更して引継ぎ
S64
テストパラメタ情報型変更して引継ぎ
S65
デフォルト値で新規生成
処理2 END

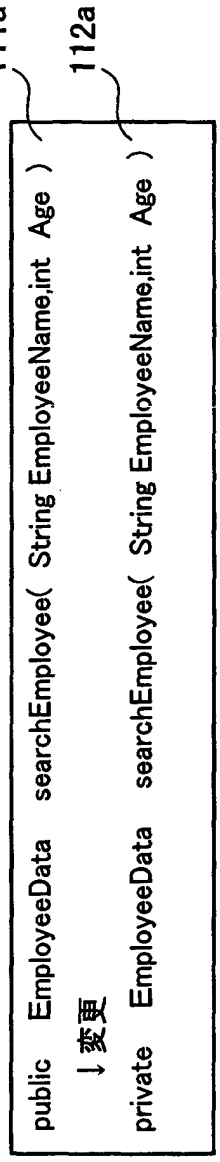
【図２２】

	メソッド名	パラメタ	戻り値
第1の判定	○	○	○
第2の判定	○	○	×
第3の判定	○	×	○
第4の判定	○	×	×
第5の判定	×	○	○
	×	○	×

【図２３】

例名	判定	引き継がれたテストケース	引き継がれなかったテストケース
メソッド1	判定1:3属性完全一致	M1-1～M1-3	なし
メソッド2	判定2:戻値不一致	M2-1～M2-7	なし
メソッド3	判定3: パラメタ不一致	M3-1～M3-10	なし
メソッド4	判定4: パラメタ、戻り値 不一致	M4-1～M4-6	なし
メソッド5	判定5: パラメタ名 不一致	M5-1～M5-10	なし
メソッド6	判定で引き継ぎできず	なし	M6-1～M6-4

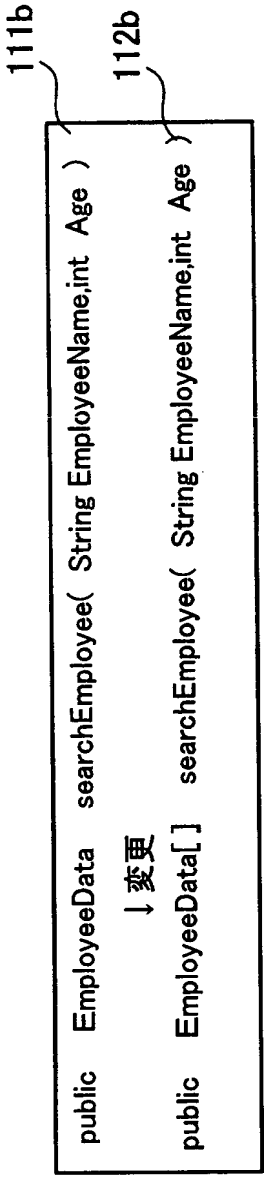
【図２４】


111a
112a
public EmployeeData searchEmployee(String EmployeeName,int Age)
↓変更
private EmployeeData searchEmployee(String EmployeeName,int Age)

【図２５】

	メソッド名	パラメタ	戻り値	その他
変更前	searchEmoployee	String EmployeeName, int Age	EmployeeData	Public
変更後	searchEmoployee	String EmployeeName, int Age	EmployeeData	private

【図２６】


111b
112b
public EmployeeData searchEmployee(String EmployeeName,int Age)
↓変更
public EmployeeData[ ] searchEmployee(String EmployeeName,int Age)

【図２７】

	メソッド名	パラメタ	戻り値	その他
変更前	searchEmoployee	String EmployeeName,	EmployeeData	Public
変更後	searchEmoployee	String EmployeeName,	EmployeeData[ ]	public

【図２８】

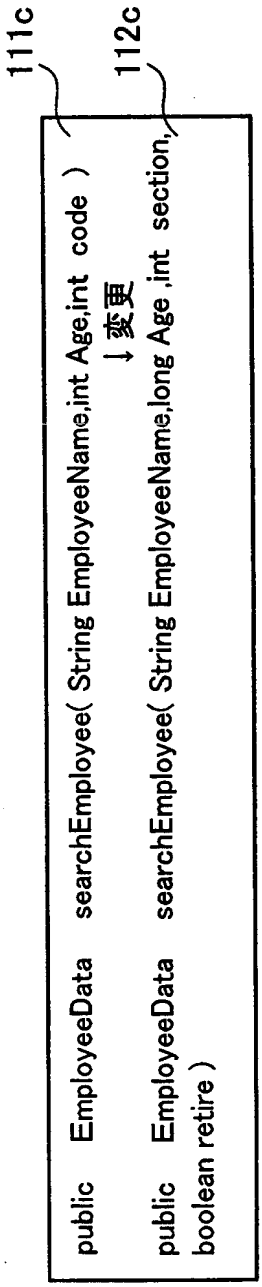
↓デフォルト値(Null)で設定　　152b

No	メソド名	パラメタ		正常/異常	戻り値	テストの観点
	searchEmployee	EmployeeName	Age		Employee Data[ ]	
M2-1	searchEmployee	“F通　太郎”	29	正常	検索結果	正常系のテスト
M2-2	searchEmployee	“”	29	異常	Null	氏名が空白の場合の異常系
M2-3	searchEmployee	Null	29	異常	Null	氏名がnullの場合の異常系
M2-4	searchEmployee	“F通　花子”	0	異常	Null	年齢が0の場合の異常系
M2-5	searchEmployee	“F通　花子”	10	異常	Null	年齢が従業員雇用範囲外の場合
M2-6	searchEmployee	“F通　花子”	65	異常	Null	年齢が従業員雇用範囲外の場合
M2-7	searchEmployee	“F通　花子”	Null	異常	Null	年齢がnullの場合

特２００２－２７８８０６

出証特２００３－３０１３４９７

【図２９】


111c
112c
public EmployeeData searchEmployee(String EmployeeName,int Age,int code)
↓変更
public EmployeeData searchEmployee(String EmployeeName,long Age ,int section,
boolean retire)

【図３０】

	メソッド名	パラメタ	戻り値	その他
変更前	searchEmoployee	String EmployeeName, int Age, int code	EmployeeData	Public
変更後	searchEmoployee	StringEmployeeName, long Age,int section, boolean retire	EmployeeData	Public

【図３１】

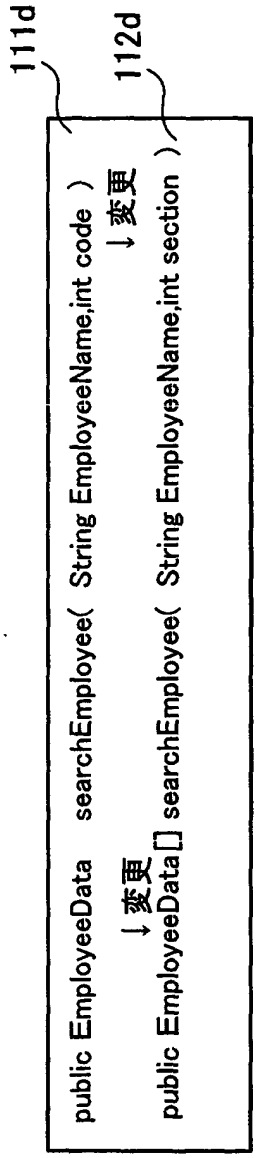
No.	メソッドパラメタ				正常/異常	戻り値	テスト観点
前	EmployeeName (String)	Age (int)	Code(int)	なし			
判定	①両方一致	②名前一致..	③型一致	④不一致(デフォルト生成)			
後	EmployeeName (String)	Age (long)	Section (int)	Retire (Boolean)			
M3-1	“F通　太郎”	29	12345	true	正常	検索結果	正常系のテスト
M3-2	“”	29	12345		異常	Null	氏名が空白の場合の異常系
M3-3	Null	29	12345		異常	Null	氏名がnullの場合の異常系
M3-4	“F通　花子”	0	12345		異常	Null	年齢が0の場合の異常系
M3-5	“F通　花子”	10	12345		異常	Null	年齢が従業員雇用範囲外の場合
M3-6	“F通　花子”	65	12345		異常	Null	年齢が従業員雇用範囲外の場合
M3-7	“F通　花子”	Null	12345		異常	Null	年齢がnullの場合
M3-8	“F通　花子”	29	0		異常	Null	部コードが0場合
M3-9	“F通　花子”	29	Null		異常	Null	部コードがnullの場合
M3-10	Null	Null0			異常	Null	氏名、年齢がnullの場合

152c

出証特２００３－３０１３４９７

特２００２－２７８８０６

【図３２】


111d
112d
public EmployeeData searchEmployee(String EmployeeName,int code)
↓変更
public EmployeeData[] searchEmployee(String EmployeeName,int section)

【図３３】

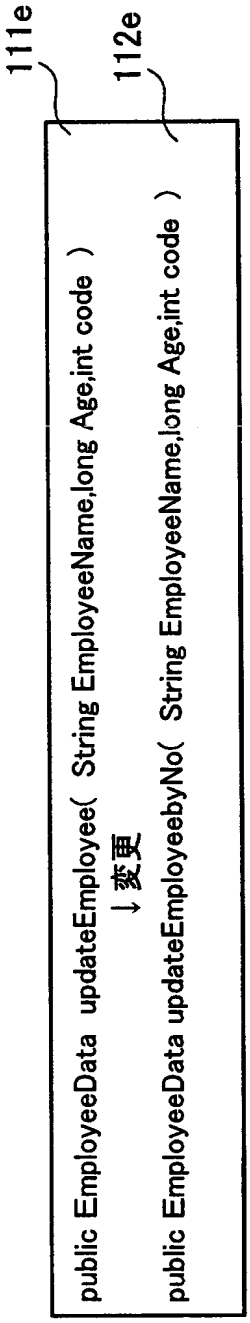
	メソッド名	パラメタ	戻り値	その他
変更前	searchEmoployee	String EmployeeName, int code	EmployeeData	Public
変更後	searchEmoployee	String EmployeeName, int section	EmployeeData[]	Public

【図３４】

152d

No	メソッドパラメタ		正常 異常	戻り値	テスト観点
前	EmployeeName (String)			EmployeeData	
判定	①両方一致	③型一致			
後	EmployeeName (String)	Section (int)		EmployeeData [ ]	
M4-1	“F通 太郎”	12345	正常	Null	正常系のテスト
M4-2	“”	12345	異常	Null	氏名が空白の場合の異常系
M4-3	Null	12345	異常	Null	氏名がNullの場合の異常系
M4-4	“F通 花子”	0	異常	Null	部コードが0場合
M4-9	“F通 花子”	Null	異常	Null	部コードがNullの場合
M4-6	Null	Null	異常	Null	氏名、年齢がNullの場合

【図３５】


111e
112e
public EmployeeData updateEmployee(String EmployeeName,long Age,int code)
↓変更
public EmployeeData updateEmployeebyNo(String EmployeeName,long Age,int code)

【図３６】

	メソッド名	パラメタ	戻り値	その他
変更前	updateEmoployee	String EmployeeName, long Age, int code	EmployeeData	Public
変更後	updateEmployeebyNo	String EmployeeName, long Age, int code	EmployeeData	Public

【図３７】

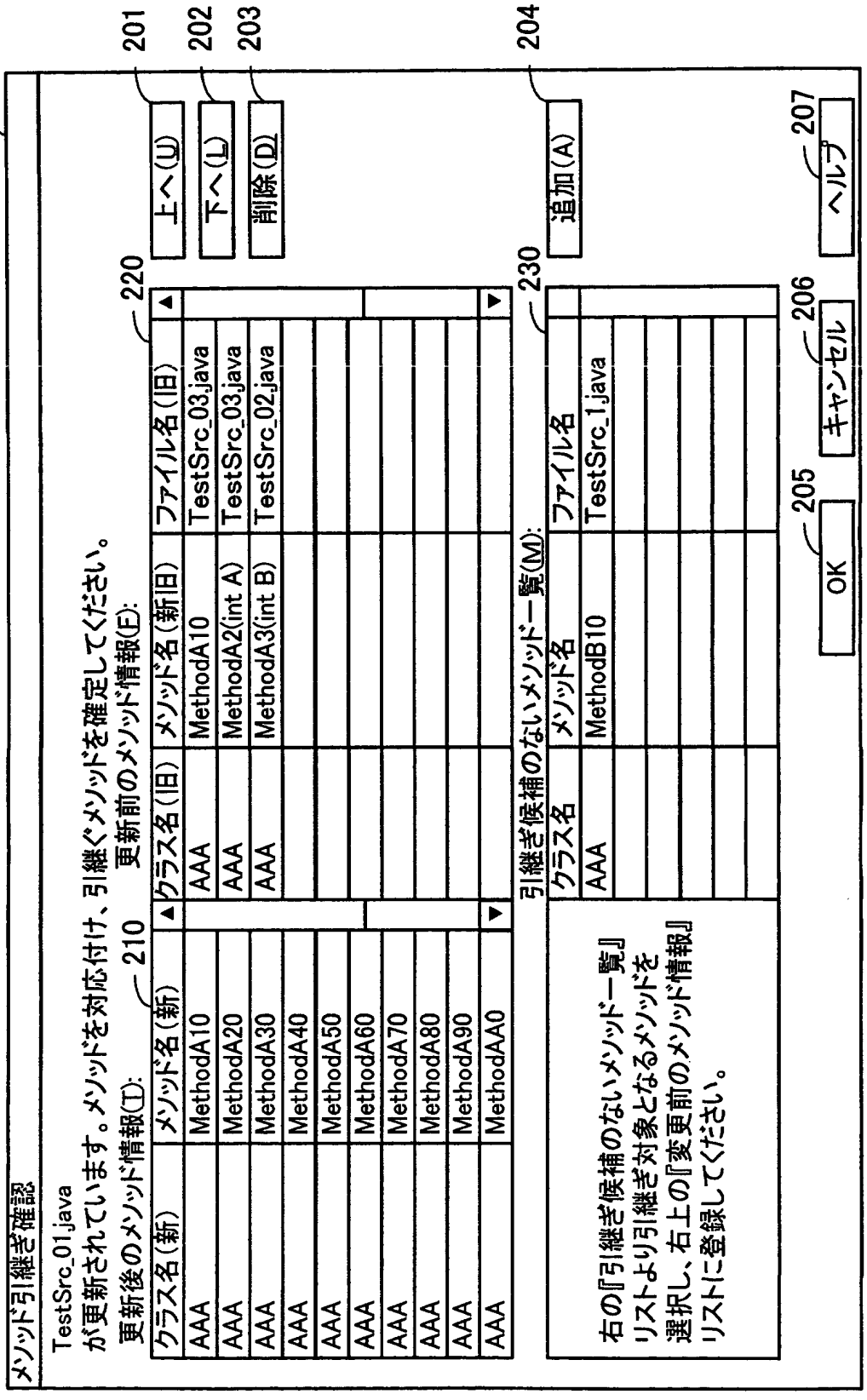
152e

No	メソッド名	パラメタ			正常/異常	戻り値	テストの観点
	updateEmployebyNo	EmployeeName	Age	code			
M5-1	updateEmployebyNo	“F通 太郎”	29	12345	正常	検索結果	正常系のテスト
M5-2	updateEmployebyNo	“”	29	12345	異常	Null	氏名が空白の場合の異常系
M5-3	updateEmployebyNo	Null	29	12345	異常	Null	氏名がNullの場合の異常系
M5-4	updateEmployebyNo	“F通 花子”	0	12345	異常	Null	年齢が0の場合の異常系
M5-5	updateEmployebyNo	“F通 花子”	10	12345	異常	Null	年齢が従業員雇用範囲外の場合
M5-6	updateEmployebyNo	“F通 花子”	65	12345	異常	Null	年齢が従業員雇用範囲外の場合
M5-7	updateEmployebyNo	“F通 花子”	Null	12345	異常	Null	年齢がNullの場合
M5-8	updateEmployebyNo	“F通 花子”	29	0	異常	Null	従業員コードが0の場合
M5-9	updateEmployebyNo	“F通 花子”	29	Null	異常	Null	従業員コードがNullの場合
M5-10	updateEmployebyNo	Null	Null	Null	異常	Null	従業員コード、氏名、年齢がNullの場合

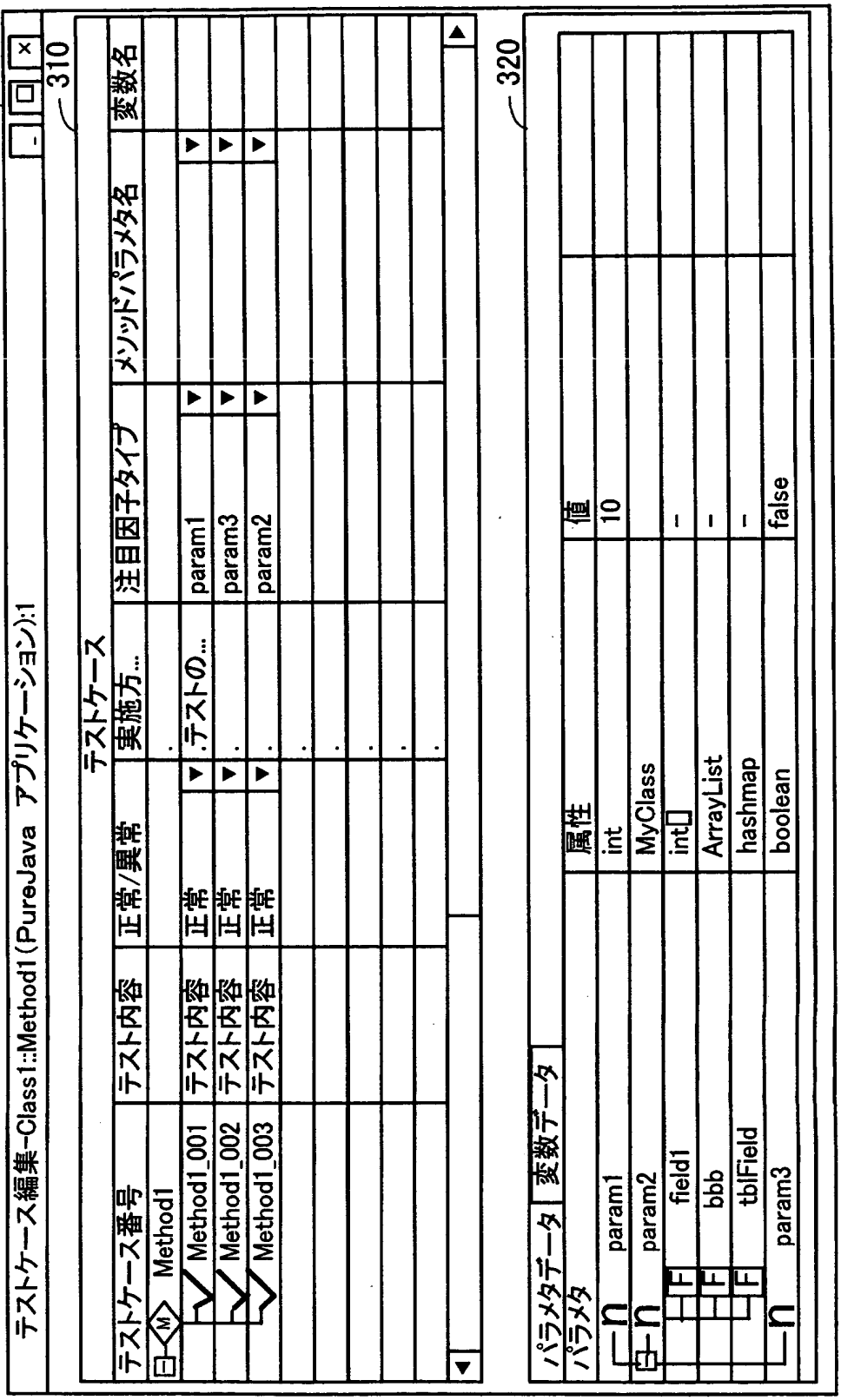
出証特２００３－３０１３４９７

特２００２－２７８８０６

【図３８】


200
メソッド引継ぎ確認
TestSrc_01.java
が更新されています。メソッドを対応付け、引継ぐメソッドを確定してください。
更新後のメソッド情報(E):
210
更新前のメソッド情報(E):
220
201
202
203
204
上へ(U)
下へ(L)
削除(D)
追加(A)
クラス名(新)
メソッド名(新)
クラス名(旧)
メソッド名(新旧)
ファイル名(旧)
AAA
MethodA10
AAA
TestSrc_03.java
AAA
MethodA20
AAA
MethodA2(int A)
TestSrc_03.java
AAA
MethodA30
AAA
MethodA3(int B)
TestSrc_02.java
AAA
MethodA40
AAA
MethodA50
AAA
MethodA60
AAA
MethodA70
AAA
MethodA80
AAA
MethodA90
AAA
MethodAA0
引継ぎ候補のないメソッド一覧(M):
230
クラス名
メソッド名
ファイル名
AAA
MethodB10
TestSrc_1.java
右の『引継ぎ候補のないメソッド一覧』
リストより引継ぎ対象となるメソッドを
選択し、右上の『変更前のメソッド情報』
リストに登録してください。
205
OK
206
キャンセル
207
ヘルプ

【図３９】


300
テストケース編集-Class1::Method1(PureJava アプリケーション):1
310
テストケース
テストケース番号
テスト内容
正常/異常
実施方...
注目因子タイプ
メソッドパラメタ名
変数名
Method1
Method1_001
テスト内容
正常
.テストの...
param1
Method1_002
テスト内容
正常
param3
Method1_003
テスト内容
正常
param2
320
パラメタデータ
変数データ
パラメタ
属性
値
param1
int
10
param2
MyClass
field1
int[]
-
bbb
ArrayList
-
tblField
hashmap
-
param3
boolean
false

【書類名】　　要約書

【要約】

【課題】　テスト対象のプログラムの内容が更新された場合に、更新後のプログラム用のテストパターンを効率的に作成できるようにする。

【解決手段】　開発対象プログラムの内容が更新されたとき、更新後の開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述１を取得する（ステップＳ１）。次に、更新前の開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述２から、更新後動作記述と共通性の高い更新前動作記述２ａを選択し（ステップＳ２）、更新前の開発対象プログラムの動作テストのために作成された複数の更新前テストパターン３から、選択された更新前動作記述の動作テストのための更新前テストパターン３ａを抽出し（ステップＳ３）、抽出した更新前テストパターン３ａの少なくとも一部を引き継いで、更新後動作記述１の動作テストのための更新後テストパターン４を生成する（ステップＳ４）。

【選択図】　　図１

出証特２００３－３０１３４９７

出　願　人　履　歴　情　報

識別番号　　〔０００００５２２３〕

１．変更年月日　　１９９６年　３月２６日
［変更理由］　住所変更
住　所　神奈川県川崎市中原区上小田中４丁目１番１号
氏　名　富士通株式会社